COMPRESSED VIDEO INDEXING AND RETRIEVAL SYSTEM USING TWO DIMENSIONAL REPRESENTATION OF COLOR

by

M. Anjaneya Prasad

DEPARTMENT OF ELECTRICAL ENGINEERING

INDIAN INSTITUTE OF TECHNOLOGY, KANPUR

June 2004

Compressed Video Indexing and Retrieval System using Two Dimensional Representation of Color

A Thesis Submitted

in Partial Fulfillment of the Requirements

for the Degree of

Master of Technology

by

M. Anjaneya Prasad

to the

DEPARTMENT OF ELECTRICAL ENGINEERING

INDIAN INSTITUTE OF TECHNOLOGY, KANPUR

June 2004

CERTIFICATE

It is certified that the work contained in the thesis entitled "*Compressed Video Indexing and Retrieval System using Two Dimensional Representation of Color*" by *M. Anjaneya Prasad* has been carried out under my supervision and that this work has not been submitted elsewhere for a degree.

June 2004

Dr. Sumana Gupta
Professor,
Department of Electrical Engineering,
Indian Institute of Technology,
Kanpur-208016.

Acknowledgements

At first, I would like to express my sincere gratitude to my thesis supervisor, Dr. Sumana Gupta for her continuous encouragement and invaluable advices. Throughout my thesis, she has been patient enough to listen carefully to my problems, provide suggestions and relevant reference materials. Without her help, it would not have been possible for me to complete my thesis in time and also to achieve success in my personal life.

I would like to thank the professors in Electrical Engineering Department, who have helped me to develop the basic understanding in signal-processing and communication subjects. I also thank the office staff members for the help I have received during my stay here.

I am thankful to Tej for his suggestions regarding implementation issues. Special thanks goes to Baju for the discussions we had throughout my thesis. I would like to thank Swagat for his help during documentation. I am thankful to Dinesh, Ravi, Bapi, Behera and Padlikar for their support during my stay over here.

I am indebted to my parents and brother for their constant support and encouragement. Last but not least I owe to God. Without his blessings nothing would have been possible in my life.

Anjaneya Prasad

Abstract

One of the challenging problems in creating multimedia database is the organization of the visual information. Since video requires large amounts of storage and processing, efficient indexing and retrieval of video has become a necessity. Content based video indexing and retrieval systems use visual features like color. Color based video indexing and retrieval methods proposed so-far use 3D representation of color. In this thesis, we propose a new method for indexing and retrieval of video using 2D representation of color. This representation reduces the retrieval time significantly. A method of mapping of color from three dimensional space to two dimensional space is discussed. Video indexing tools developed support automatic segmentation of video, identification of keyframes, keyframe clustering and extraction of visual features. These visual features are used for efficient video retrieval. The proposed video indexing method uses DC frames of $MPEG$ compressed bit streams. Abrupt scene changes as well as special editing effects such as dissolves and fades are detected accurately. A new method for keyframe clustering is proposed which reduces the redundancies in the keyframes. Color layout descriptor is used to extract the indices of the keyframes and to retrieve the video segments efficiently. Experimental results obtained prove that retrieval time is significantly reduced using the proposed 2D representation of color.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Need for video indexing

Multimedia information systems are becoming increasingly important with the advent of broadband networks, high-powered work stations and compression standards. Information databases have evolved from simple text to multimedia with video, audio and text. Since visual media requires large amounts of storage and processing, there is a need to efficiently store, index and retrieve the visual information from multimedia database. In this contextual framework, indexing of video is similar to text indexing or book marking while retrieval or searching of this video snaps or objects is similar to paragraph searching in the conventional textual database framework.

Video indexing basically involves segmentation of video into identifiable partitions called shots by determining the positions of significant scene changes. It also involves the determination and classification of various video editing effects like dissolves and fades, and selection of representative frames(keyframes) to represent each shot. Indexing also involves keyframe clustering, which reduces the redundancies in keyframes. These redundancies in keyframes are due to visual similarities in different shots of a video. These keyframes form the basis for indexing each video partition(shot). Further operations such as, video browsing and content-based retrieval of video can then be performed using, only these keyframes, instead of the entire video.

Video sequences are usually stored and transmitted in compressed format. Hence video indexing can be done in the compressed domain itself or the sequence can first be decompressed and then used for video indexing. Working in compressed domain has the following advantages: less computational complexity, lower storage space and operations are faster.

1.2 Retrieval

One of the main application of video indexing is to search the video by its content. The user poses queries to the large database of videos and a fast, efficient, and precise reply is required. In a typical query, the user provides an image or another video and expects the system to retrieve similar clips (i.e. a "query-by-example"). Perhaps, the most challenging questions in video retrieval are: "what are the good sets of features to represent the video?" and "what is the good measure of visual similarity?" The overall system of video indexing and retrieval are shown in Fig.1.1

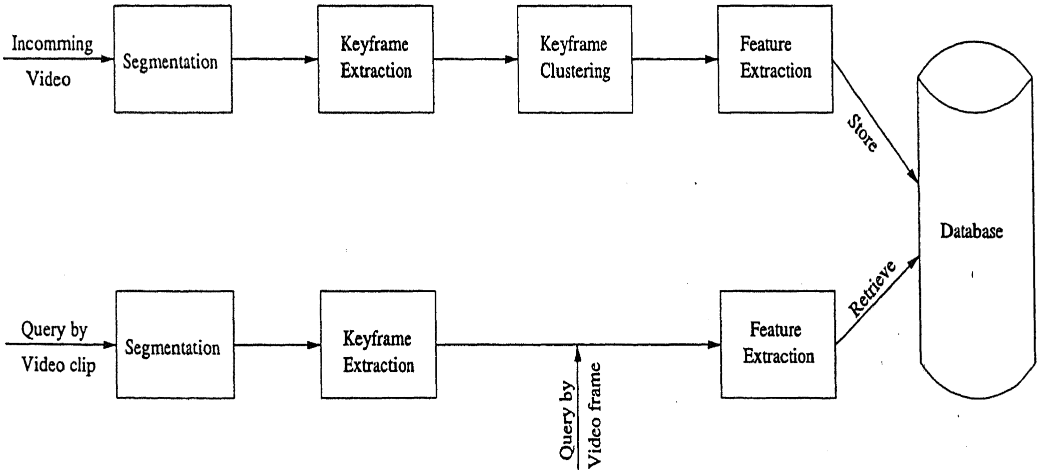


Figure 1.1: Block diagram of indexing and retrieval

1.3 Existing color indexing approaches

Content-based video indexing and retrieval systems use features like color, texture and motion information. Color is one of the most widely used visual features in image and video retrieval. Color features are relatively robust to changes in the background colors and are independent of image size and orientation. In this thesis we have used color for indexing and retrieval.

One common method of characterizing color is to use color histograms. Ferman *et al* [2] proposed histogram based color descriptors, to capture the color properties of multiple images or group of frames. They proposed alpha-trimmed average histograms and intersection histograms to represent group of frames. Kasutani and Yamada [3] proposed color layout descriptor for image/video retrieval. This descriptor describes the spatial distribution of colors in keyframes. In [4] region segmentation-based projective histograms and its moments are used as database indices. In [5] color and texture descriptors for image/video indexing are described. Dominant color descriptor describes the global as well as local spatial color distribution in images for high-speed retrieval and browsing.

1.4 Objective of the Thesis

The major problem of most of the indexing techniques is the higher dimensionality of feature space. Lower dimensional indexing reduces the storage cost of indices and increases the retrieval speed. The methods proposed so-far for indexing and retrieval uses three dimensional(3D) representation of color. If the dimensionality of color can be reduced to two then the dimensionality of feature space can be reduced and hence retrieval time can be reduced.

The main objective of this thesis is to develop a content based video indexing and retrieval system using two dimensional representation of color. To achieve this, the first step is to represent the three dimensional RGB color space in two dimensional plane. In [1] a two dimensional representation of color(YC) is proposed for compression applications. We propose an indexing scheme to index videos in this space. Scene change detection algorithm of [13] is modified, such that both abrupt scene changes and gradual scene changes can be detected using the same data extracted form the DC frames of $MPEG$ compressed video. A new method of keyframe clustering is proposed to reduce the redundancies in keyframe database. Color layout descriptor proposed in [3] is used for indexing the keyframes and retrieval of video segments.

1.5 Organization of the Thesis

The first step in our work is to obtain a two dimensional representation of color. In chapter 2 the important properties of RGB and YUV color planes are discussed. Based on these properties a spiral approximation is discussed to obtain a single color signal. The image is transformed from RGB to YUV color space, and the two chroma signals (U & V) are combined to obtain the single color(C) signal. Segmentation of MPEG compressed video is detailed in chapter 3. Segmentation methods of uncompressed and compressed videos are reviewed, and proposed method of scene change detection is described in this chapter. Chapter 4 discusses the method used for keyframe selection, need for keyframe clustering, and also describes a new method for keyframe clustering. Chapter 5 describes the method of indexing keyframes and also discusses the retrieval of video segments. Chapter 6 reports the results of various experiments. Finally chapter 7 concludes the thesis with suggestions for future work.

Chapter 2

The Color Plane and Spiral Map

The study of color is important in the design and development of the color vision system. Use of color in image displays is not only more pleasing, but it also enables one to distinguish between thousands of colors. The color representation is based on the classical theory of Thomas Young (1802), who stated that any color can be reproduced by mixing a set of three primary colors. The RGB color assumes the red, green and blue as primary colors.

The RGB color space is the basic choice for computer graphics and image processing and these are the primary additive colors. However RGB is not very efficient for real world images as transmission primaries, since equal bandwidths are required to describe all the three colors. So, in 1950s National Television Systems Committee (NTSC) developed the YIQ system for transmission of color images, using the existing monochrome channels without increasing the bandwidth and maintaining compatibility with monochrome television system. The YIQ system takes advantage of human visual system's greater sensitivity to luminance than chrominance components. YUV is similar color representation used for PAL and SECAM systems. The YUV has an important properties that facilitates the spiral mapping of $U - V$ plane. These will be discussed in the following sections.

2.1 Color Representation

The perceptual attributes of color are *brightness, hue* and *saturation.* Brightness represents the perceived luminance. The hue of a color refers to its 'redness', 'greenness' and so on. For monochrome light sources, differences in hues are manifested by the difference in wavelengths. Saturation is that aspect of the perception that varies most strongly as more white light is added to a monochrome light. The Fig.2.1 shows a perceptual representation of the color space. Brightness varies along the vertical axis, hue (θ) varies along the circumference, and saturation (S) varies along the radial distance. For fixed brightness W, the symbols R, G and B show the relative locations of red, green and blue spectral colors.

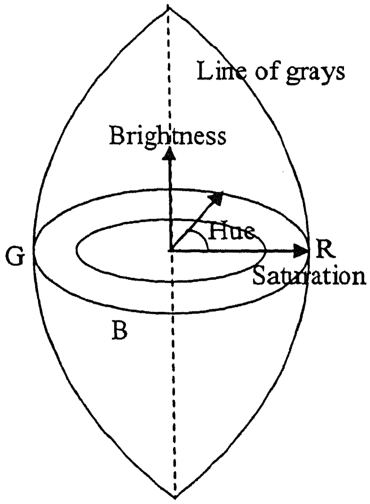


Figure 2.1: Perceptual Representation of Color Space

Luminance is an objective measure of that aspect of visible radiant energy that produces the sensation of brightness. Radiation of different wavelengths contributes differently to the sensation of brightness. We can also conclude that Y signal gives the luminance or light intensity as perceived by the eyes, regardless of the color of the object seen. The contribution of the primary colors for obtaining Y is given by

$$Y = (0.299)R + (0.587)G + (0.114)B \tag{2.1}$$

Hue is the predominant spectral color of the received light. The color of any object is distinguished by its hue or tint, like red, green, blue, brown, orange and so on. The spectrum of such signals lies along different phase angles of the color planes. In other words, hue is another name for the phase angle of the color plane which is discussed in next section.

When a pure spectral color is mixed with white color, the color of the mixture will be less in purity compared to actual color, but at the same time the brightness of the mixture increases. The study of such experiments on color planes reveals that the purest colors have the maximum radius and when white light is mixed the radius in the color planes decreases i.e. the purity of color decreases.

In three dimensional RGB space, the RGB values of the colors give the co-ordinates of the corners of rectangular block as shown in Fig.2.2. The six faces of this block are planes given by $R = 0, G = 0, B = 0$, and $R = 1, G = 1, B = 1$ and they enclose all the points in RGB space which have valid combinations of R, G and B.

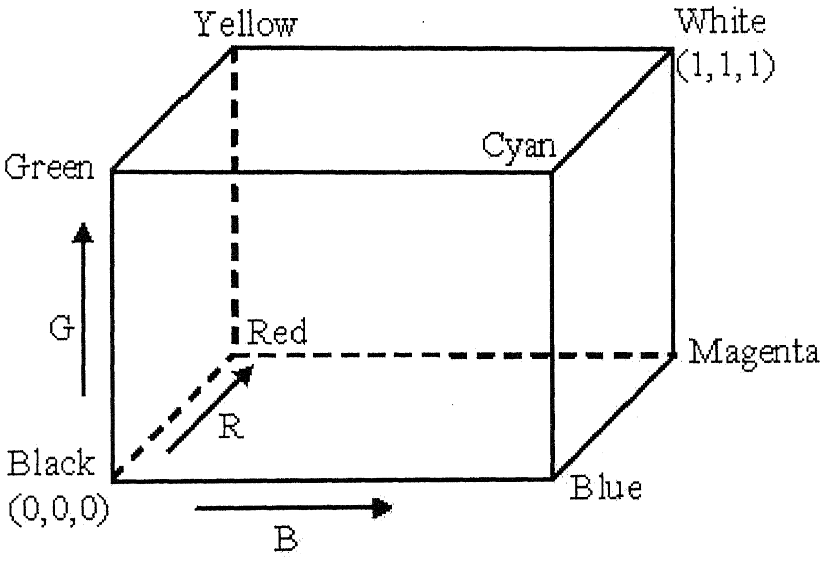


Figure 2.2: Valid Colors of RGB Color Space

By transforming the RGB points on the surface of this block into YUV values, a block can be constructed in 3-D YUV space which encloses all YUV values corresponding to all valid RGB values. This block will be referred to as the 'YUV valid–Color' block and its shape is shown in Fig.2.3.

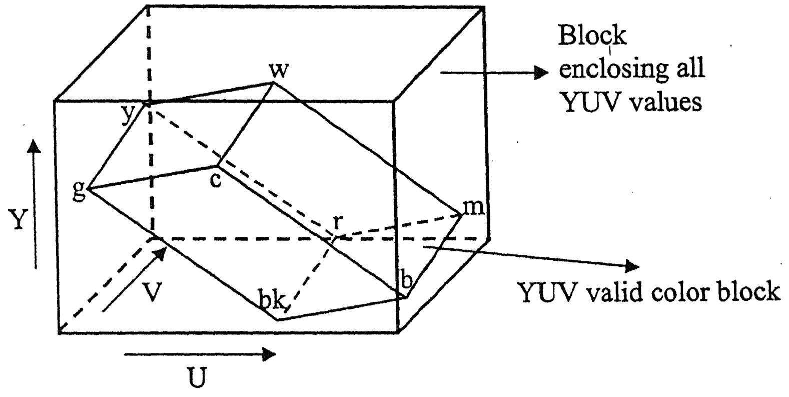


Figure 2.3: Color Space Transformed From RGB to YUV

$$\begin{aligned} Y &= 0.299R + 0.587G + 0.114B \\ U &= -0.299R - 0.587G + 0.889B \\ V &= 0.701R - 0.587G - 0.114B \end{aligned} \qquad (2.2)$$

Since the RGB to YUV transformation is a linear process, the corners of the YUV valid–color are like the RGB block of Fig.2.3 given by the co-ordinates of the colors, and the surfaces are flat planes corresponding to $R = G = B = 0$ and $R = G = B = 1$. Further details of the shape of this YUV block can be conveniently illustrated by three, mutually perpendicular, views or projections obtained by viewing along the Y, U and V axes. These views are obtained by plotting U versus V, Y versus V, and Y versus U respectively as shown in Fig.2.4.

In Fig.2.4, the four saturated colors having $B = 0$ (black, red , green and yellow) and those having $B = 1$ (blue, magenta, cyan and white) lie along straight lines in the Y-U plane.

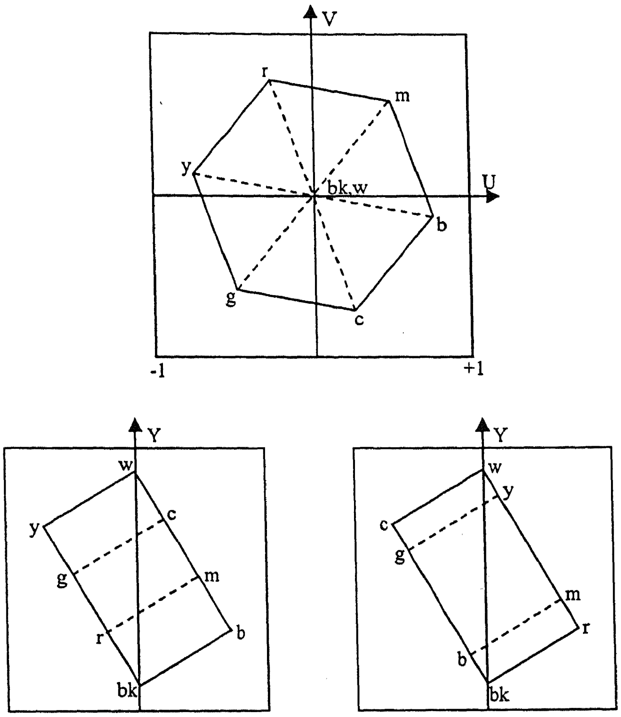


Figure 2.4: YUV Color Space Along Different Axis

This implies that the faces of the valid-color block given $B = 0$ and $B = 1$ are perpendicular to the $Y - U$ plane. Similarly, the faces given by $R = 0$ and $R = 1$ are perpendicular to the $Y - V$ plane.

2.2 Properties of Color Plane

The color plane possesses very important properties. It can be observed from Fig.2.5 that the complementary pairs (Blue-Yellow, Red-Cyan, and Green-Magenta) fall exactly at 180^0 out of phase and the entire color plane is nearly split into six equal regions, between two adjacent major color lines like Blue-Magenta, Magenta-Red etc [1].

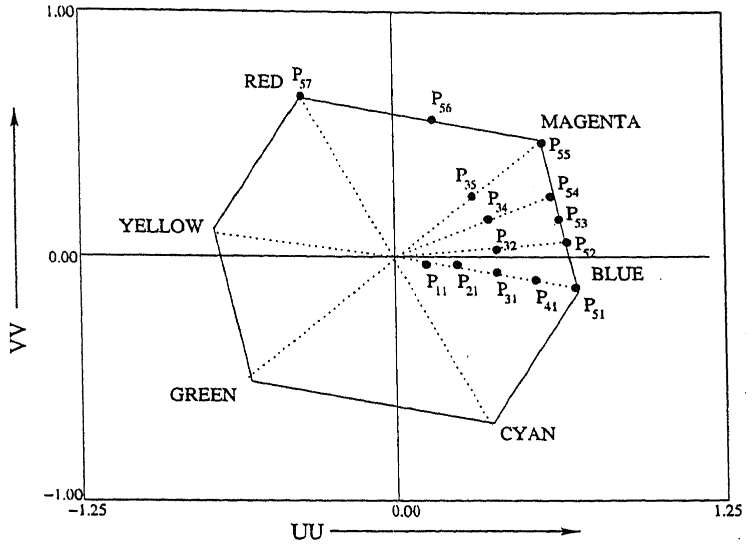


Figure 2.5: Color Plane

2.2.1 Behavior Along Radius

For any chosen angle, the change in the primary colors for that angle is linear and systematic. Let us consider the case of blue color line, and choose the points at 0%, 25%, 50%, 75% and 100% radii designated by $P_{11}, P_{21}, P_{31}, P_{41}$ and P_{51} respectively in Fig.2.5.

%	Poi-nts	Y	U	V	R	G	B	Min (RGB)	R'	G'	B'
0	P_{11}	0.5	0.0	0.0	0.5	0.5	0.5	0.5	0.0	0.0	0.0
25	P_{21}	0.5	0.2215	-0.0285	0.4715	0.4715	0.7215	0.4715	0.0	0.0	0.25
50	P_{31}	0.5	0.443	-0.057	0.443	0.443	0.943	0.443	0.0	0.0	0.5
75	P_{41}	0.5	0.6645	-0.0855	0.4145	0.4145	1.1645	0.4145	0.0	0.0	0.75
100	P_{51}	0.5	0.886	-0.114	0.386	0.386	1.386	0.386	0.0	0.0	1.0

Table 2.1: The comparison of all components along radial direction

For all the calculations, we assume that Y=0.5. The 100% pure blue color point P_{51} is given by $U = 0.886, V = -0.114$. For other points P_{11} to P_{41}, percentage of the radius along the blue line is taken and the values for U and$V$ are obtained.

In Table 2.1 $min(R, G, B)$ is the minimum value among the R, G, B values, and

$$\begin{aligned} R^{'} &= R - min(R, G, B) \\ G^{'} &= G - min(R, G, B) \\ B^{'} &= B - min(R, G, B) \end{aligned} \tag{2.3}$$

We can observe from the Table 2.1, that for point P_{11}, the U and V values equal zero and the point is actually on the origin (achromatic point). The $min(R, G, B)$ value is equal to 0.5, which is also the value of Y. In other words $min(R, G, B)$ is the quantity, which exists for all the colors, and contributes only to the Y values along that axis of the color plane.

The remaining color quantities span the color plane and are given by the $R^{'}, G^{'}$ and $B^{'}$. These are obtained by subtracting $min(R, G, B)$ from R, G, B values.

Now considering the $(R^{'}, G^{'}, B^{'})$ values as given in Table 2.1, we note $B^{'}$ changes linearly with the change in radius. Similarly, in the other directions, the changes along the radii are linear. However the ratio among $R^{'}, G^{'}, B^{'}$ changes with angle, as discussed below.

2.2.2 Behavior Along Phase

The points P_{51} and P_{55} are the pure blue and magenta colors respectively. The distance between these two points is split into three equal parts, and we get the points P_{52} and P_{54} as shown in Fig.2.5. For convenience, we would like to handle half radii points in the respective directions. These are shown by the points $P_{31}, P_{32}, P_{34}, P_{35}$.

Once again keeping the value of Y fixed at 0.5 and varying the U and V of these points, we obtain the $R^{'}, G^{'}, B^{'}$ values as listed in Table 2.2.

We observe that $G^{'}$ and $B^{'}$ remains constant at 0.0 and 0.5, while $R^{'}$ changes linearly from 0.0 and 0.5. Hence the ratio between the color values changes with the change in phase angle. To make this change linear w.r.t. the phase angle, the maximum radius should be

Points	Y	U	V	R	G	B	$Min'(R,G,B)$	R'	G'	B'
P_{31}	0.5	0.443	-0.057	0.443	0.443	0.943	0.443	0/6	0.0	0.5
P_{32}	0.5	0.393	0.06	0.560	0.393	0.893	0.393	1/6	0.0	0.5
P_{34}	0.5	0.343	0.177	0.677	0.343	0.843	0.3435	2/6	0.0	0.5
P_{35}	0.5	0.293	0.293	0.793	0.293	0.793	0.2935	3/6	0.0	0.5

Table 2.2: Comparison of various components along phase angle

made equal in all directions. To achieve the same with unit radius, the U and V axes of color plane are multiplied by $1/0.9$ and $1/0.7$ respectively. Such a modified color plane is shown in Fig.2.6, with new axes as UU and VV respectively.

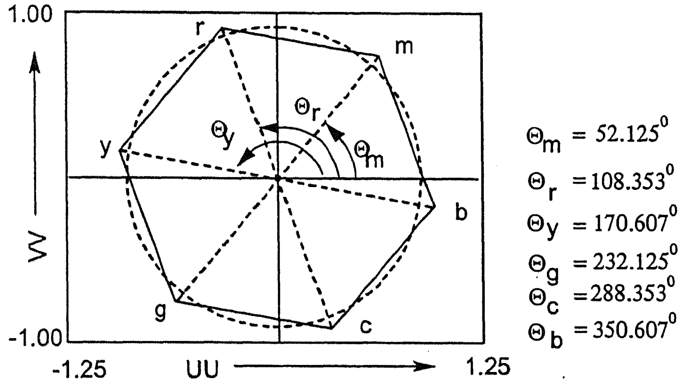


Figure 2.6: Modified Color Plane

2.2.3 The Color Gamuts

The Y signal lies along the $z - axis$ of the color planes and the actual signal space is three dimensional space. The color planes that we refer to are the planes corresponding to top view of color space. Along the cross-section of this color space for different values of Y, we get the color gamut for a particular value of Y. The shape of such gamuts varies in regular fashion as Y varies from 0.0 to 1.0 [1].

2.3 Spiral mapping of Color Plane

In [1] a method to combine two chrominance signals by spiral mapping was suggested. The two chrominance signals shared many similar properties that are utilized for approximation in the color plane. In this thesis we utilize the approximation for efficient video indexing and retrieval. By indexing in approximated two dimensional YC [1] space we can reduce the length of feature vector and also the number of indices per keyframe, as compared to indexing in $3D$ $YCbCr$ space discussed in [3].

2.3.1 Spiral Approximation

It is noted from above observations that *saturation* varies along the radius in the color plane, and *hue* varies along the phase in the color plane. As human visual perception is more sensitive to hue(*angle*) than saturation(*radius*), a change in radius is more tolerable than a change in phase angle. So approximating radius to some other value is acceptable as long as the phase angle is preserved. These points are exploited in arriving at spiral mapping of color plane. The luminance signal Y in YUV is preserved, and the color plane spanned by U and V is spiral mapped to get a single color signal that represents both U&V. In this section a method for approximating the color plane by spiral is discussed.

2.3.2 The Nature of Spiral

The spiral for five encirclements of the color plane is as shown in Fig.2.7. This spiral spans the whole color plane, if the radius is made unity in all directions of color plane, hence the modified color plane's radius is taken as unity. Number of encirclements of spiral is denoted by L, and the radius of any point on the spiral is denoted by C.

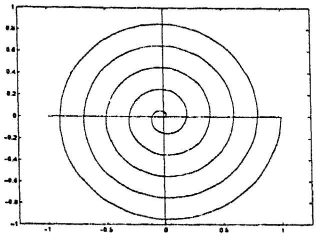


Figure 2.7: Spiral with L =5 on Color Plane

The value of any spiral point is given by the radius C of that point on the spiral. Therefore for any given value of L, with $C = 0.0$ origin, the spiral point at the completion of the first circle will have the value equal to $1.0/L$ i.e. $C = 1.0/L$, similarly at completion of second circle, $C = 2.0(1.0/L)$ and so on. Finally at the completion of L^{th} circle, $C = L(1.0/L) = 1.0$.

We can calculate exact phase information from the spiral signal itself. This is illustrated in the following example

$$\text{Suppose } L = 5 \text{ and } C = 0.42$$
$$CL = (5) \times (0.42) = 2.1$$

The point lies above two circles, i.e. on the third circle of the spiral and the exact phase angle is obtained by ignoring the integer part from 2.1, i.e. 2.1-2.0=0.1. It implies that the point lies at a phase given by

$$(0.1) \times (360) = 36^0 \text{ (or) } 10\% \text{ of } 360^0$$

For a given phase ϕ, it can have any one of the L values of the radii, For example, suppose $L = 5$ and $\phi = 45^0$, the possible C values are

$$C_1 = \frac{0+(45/360)}{L} = 0.025$$
$$C_2 = \frac{1+(45/360)}{L} = 0.225$$

$$\dots \dots \dots$$

$$C_L = \frac{(L-1)+(45/360)}{L} = 0.825$$

2.3.3 How to Map U-V Plane Onto Spiral ?

Radius(r) and phase(θ) of points in the color plane(U-V) are defined as follows:

$$r = \sqrt{U^2 + V^2}$$
$$\theta = tan^{-1}\frac{V}{U} \tag{2.4}$$

Points on the color plane are mapped onto spiral by approximating radius and preserving phase. Any point on the color plane is mapped to the nearest point on the spiral. This mapping is as follows: The radius values of all the points in the color plane are approximated to the concentric circles whose center is at the origin of color plane and radius values $n/L, n = 1, 2, \dots, L$, where L is the number of encirclements of the spiral. The value of angle which is given by $(\theta/(2\pi L))$ is added to this approximated radius to get the final value of C. So C can be expressed as, $C = approximated\ radius + (\theta/(2\pi L))$. This C is now a point on the spiral with radius(r) of the color point approximated to C. This way color is represented by only two quantities luminance(Y) and color(C). Mapping back from spiral color(C) to color plane(U-V) is as follows:

$$U = C.cos(2\pi(LC - \lfloor LC \rfloor)) \tag{2.5}$$
$$V = C.sin(2\pi(LC - \lfloor LC \rfloor))$$

the quantity $\lfloor LC \rfloor$ gives the integral part of the product of L and C.

2.3.3.1 Example of Mapping

Let radius $r = 0.45$ and $\theta = 270°$ and L=5, then r is approximated to 0.4 and $C = 0.4 + (\theta/(2\pi L)) = 0.55$. To remap to U-V, $\theta = 2\pi(LC - \lfloor LC \rfloor) = 270°$ and using eqn.2.5, U and V can be obtained. From this example it is clear that, spiral mapping of color points is achieved by preserving phase, and approximating radius.

Chapter 3

Segmentation of MPEG Compressed Video

Temporal video segmentation is generally accepted as the first step in content based analysis of video sequences. It breaks the video into a set of meaningful and manageable segments called shots that are used as basic units for indexing and annotation. Each shot is represented by one or more keyframes. The content of the shot is indexed by spatial features like color and texture extracted from the keyframes. In addition temporal features from the shot like motion, camera operations, can be used for indexing.

A shot is defined as one or more frames generated and recorded contiguously, and representing a continuous action in time and space. Video editing produces two general types of shot transitions: *abrupt* and *gradual. Abrupt* transitions are the most common and they occur over a single frame by splicing the two distinct scenes successively. *Gradual* transitions occur over multiple frames and are result of effects such as *dissolves* and *fades*. *Dissolves* show one image superimposed on the other as the frames of the first shot get dimmer and those of the second one get brighter. *Fade* is a gradual change in brightness usually resulting in or starting with a solid color.

As the characteristics of the frames before and after an abrupt transition usually differ significantly, abrupt transitions are much easier to detect than gradual transitions. The recognition of the gradual transitions is further complicated by the presence of camera operations like pans and zoom, and object movement as they exhibit temporal variation of the same order and cause false positives.

Algorithms for scene change detection exist both for uncompressed and compressed videos. In the following section a review of methods for temporal video segmentation of uncompressed video is given.

3.1 Temporal video segmentation of uncompressed video

A majority of algorithms proposed for temporal video segmentation process uncompressed video. Usually, a similarity measure between successive frames is defined. When the two frames are sufficiently dissimilar, there may be a cut. Gradual transitions are found by using cumulative difference measures and more sophisticated threshold schemes.

Zhang[6] proposed that a change between two frames can be detected by comparing the difference in the intensity values of the corresponding pixels in the two frames. His algorithm counts the number of pixels that have changed and an abrupt change is declared if the number of pixels that have changed, expressed as a percentage of the total number of pixels, exceeds a certain threshold[6]. However this technique may produce false detections since camera movements and object motion can have the same effect on a large number of pixels and hence false scene changes will be detected.

In the likelihood ratio approach[6][7] the frames are subdivided into blocks, which are then compared on the basis of the statistical characteristics of their intensity levels. Eqn.3.1 represents the formula that calculates the likelihood function λ.

$$\lambda = \left[\frac{\sigma_i + \sigma_{i+1}}{2} + \left(\frac{\mu_i - \mu_{i+1}}{2} \right)^2 \right]^2 \bigg/ \sigma_i \sigma_{i+1} \tag{3.1}$$

μ_i and μ_{i+1} are the mean intensity values for a given region in two consecutive frames and σ_i and σ_{i+1} are the corresponding variances. The number of blocks for which λ exceeds a certain threshold are counted and if this number exceeds a certain value, a scene change is declared. A subset of the blocks can be used to detect the difference between the images so as to expedite the process of block matching. This approach is better than the pixel-based approach as it increases the tolerance to noise associated with camera and object movement. It is possible that even though the two corresponding blocks are different they can have the same density function and in such cased no change is detected.

Sensitivity to camera and object movement can be further reduced by comparing the gray level histogram of the two frames[6][8]. This is because the two frames whose backgrounds differ by a small amount and which have the same amount of object motion have almost the same histogram. The histogram is given by the number of pixels belonging to each gray level in the frame. The histogram metric is given by Eqn.3.2

$$\sum_{j=1}^{G} |H_i(j) - H_{i+1}(j)| > T_k \tag{3.2}$$

where G is the number of gray levels, j is the gray value, i is the frame number, and $H(j)$ is the value of the histogram of for the gray level j. If the sum of absolute differences of corresponding values of consecutive histograms is greater than a given threshold T_k then a transition is declared.

Zabith *et al*[9] have proposed a feature-based approach for detecting sudden scene changes. During a cut, new intensity edges appear far from the location of the old edges. Similarly, old edges disappear far from the locations of new edges. By counting the number of entering and exiting edge pixels, an abrupt scene change can be identified. However, this algorithm requires edge detection in every frame, which is computationally very costly. Another limitation of this scheme is that the edge detection method does not handle rapid changes in overall scene brightness or scenes with high contrast levels.

The twin-comparison method[6] is a histogram based technique for abrupt as well as gradual transition recognition. In the first pass a high threshold T_h is used to detect abrupt transitions. In the second pass a lower threshold T_l is employed to detect the potential starting frame F_s of gradual transition. F_s is then compared to subsequent frames. This is called an accumulated comparison as during a gradual transition this difference value increases. The end frame F_e of the transition is detected when the difference between consecutive frames decreases to less than T_l, while the accumulated comparison has increased to a value higher than T_h. If the consecutive difference falls below T_l before the accumulated comparison exceeds T_h, then the potential start frame F_s is dropped and the search continues for other gradual transitions. However for most gradual transitions the frame difference falls below the lower threshold. Such transitions can not be detected using the twin-comparison technique.

Several statistical-feature based techniques have also been proposed for gradual transition detection. Alattar used quadratic behaviour of the variance to detect fading. This algorithm can only detect fade-in and fade-out where the end frames are fixed. When the sequence has considerable motion, this algorithm fails to identify fade-in and fade-out regions.

Since $MPEG$ was established as an international standard for compression of digital video, videos are increasingly stored and transmitted in compressed format. Hence, it is highly desirable to develop methods that can operate directly on the encoded stream. Working in the compressed domain offers the following advantages. Firstly, by not having to perform decoding and re-encoding, computational complexity is reduced and saving on both decompression time and storage is obtained. Secondly, operations are faster due to the lower data rate of compressed video. Last but not the least, the encoded stream already contains a rich set of pre-computed features, such as motion vectors and block averages that are suitable for temporal video segmentation. The following section gives an overview of $MPEG$ compression standard.

3.2 Overview of MPEG compression standard

The Moving Pictures Expert Group($MPEG$) standard is the most widely accepted international standard for digital video compression. $MPEG$ video is broken up into a hierarchy of layers to help with error handling, random search, editing and synchronization. The first (top) layer is known as video sequence layer. The second layer below is the group of pictures (GOP) layer. The third layer is the picture layer itself, and the layer below that is called the slice layer. Each slice consists of macroblocks (MBs), which are 16 × 16 arrays of luminance pixels, or picture data elements, with 8 × 8 arrays of associated chrominance pixels. Macroblocks are the units of motion compensated compression. These macroblocks are further divided into 8 × 8 blocks of pixels.

$MPEG$ compression uses two basic techniques: macroblock based motion compensation to reduce temporal redundancy and transform domain block-based compression to capture spatial redundancy. An $MPEG$ stream consists of three types of pictures - intra coded(I-frames), predicted(P-frames) and bi-directional(B-frames). These pictures are combined in a repetitive pattern called group of pictures(GOP). A GOP starts with an I frame. The I and P frames are referred as anchor frames. B frames appear between each pair of consecutive anchor frames. Fig.3.1 shows a typical $MPEG$ video sequence including a GOP of 12 frames: the sub-GOP size is 3.

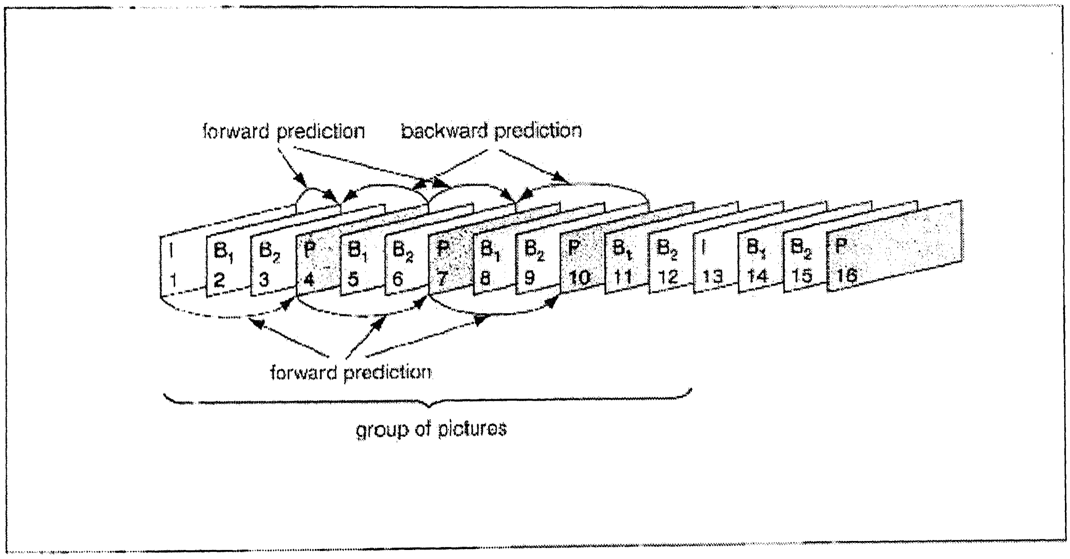


Figure 3.1: Typical MPEG compressed video sequence

Each video frame is divided into a sequence of nonoverlapping MBs. Each MB can be either intra coded or inter coded (i.e. coded with motion compensation). I frames are typically intra coded: every 8 × 8 pixel block in the MB is transformed to the frequency domain using the Discrete Cosine Transform (DCT). The first DCT coefficient is called the DC term and is 8 times the average intensity of the respective block. The 64 coefficient is then (lossy) quantized and (loss less) entropy encoded using the Run Length Encoding (RLE) and Huffman entropy encoding. As I frames are coded without reference to any other video frames, they can be decoded independently and hence provide random accesses points into the compressed video.

P frames are predictively coded with reference to the nearest past anchor frame (i.e. the previous I or P frame). For each MB in a P frame, the encoder searches the anchor frame and finds the best matching block in terms of intensity. The MB is then represented by a motion vector(MV) which points to the position of the match and the difference(residue) between the MB and its match. The residue is then DCT encoded, quantized and entropy coded while the MV is differentially and entropy coded with respect to its neighbouring MV. This is called encoding with forward motion compensation. An inter coded MB provides higher compression gain than an intra coded as the residue can be represented with fewer bits.

To achieve further compression, B frames are bi-directionally predictively encoded with forward and/or backward motion compensation with respect to the nearest past and/or future anchor frames. As B frames are not used as reference for coding other frames, they can accommodate more distortion, and thus, provide higher compression compared to I and P frames. During the encoding process a test is made on each MB of P and B frame to see if it is more expensive to use motion compensation or intra coding. The latter occurs when the current frame does not have much in common with the anchor frame(s). As a result each MB of P frame could be coded either intra or forward, while for each MB of a B frame there are four possibilities: intra, forward, backward or interpolated.

3.3 Temporal video segmentation of compressed video

Several algorithms for temporal video segmentation in the compressed domain have been reported. These algorithms use a set of pre-computed features such as DCT coefficients, DC terms, motion vectors(MV), and MB coding mode, which the encoded stream already contains.

One approach to using the DCT coefficients to find frames where camera breaks have occurred is as follows[10]. From the 8×8 blocks of a single video frame, m, that have been encoded using the DCT, a subset of blocks are chosen *apriori*. The blocks are chosen from n connected regions in each frame. Again a subset of 64 coefficients for each block is chosen. The coefficients chosen are randomly distributed among the AC coefficients of the blocks. Taking coefficients from each frame a vector is formed as follows:

$$V_m = [c_1, c_2, c_3, c_4, \ldots] \tag{3.3}$$

This vector represents the frame of the video in DCT space. The inner product is used to find the difference between the two frames:

$$\psi = \frac{V_m V_{m+1}}{|V_m||V_{m+1}|} \tag{3.4}$$

where V_m is the vector of the frame being compared and V_{m+1} is the vector of the successor frame. A transition is detected when $1 - |\psi| > t$, where t is some threshold.

Zhang *et al*[11] have also experimented with motion-based segmentation using the motion vectors in the $MPEG$ compressed data as well as the DCT coefficients. Meng *at al* have extended this concept further by performing more detailed operations on the $MPEG$ compressed data. If there is a break at B-frame, most of the motion vectors will come from the following anchor frame and few will come from the previous anchor frame. A scene cut is detected based on the ratio of the backward and forward motion vectors. When a scene change occurs at a P-frame the encoder cannot use macroblocks form the previous anchor frame for motion compensation as P-frames have only forward motion compensation. A scene break is detected based on the ratio of macroblocks without motion compensation to macroblocks with motion compensation. Since I-frames are completely intra-coded, without motion vectors, the method using DCT coefficients described above[10] can be used for scene change detection.

The algorithm we propose for scene change detection operates on the DC sequence extracted form the $MPEG$ compressed video. The following section describes the extraction of DC sequences from $MPEG$ compressed video.

3.4 Extraction of DC sequences from MPEG compressed video

A reduced image obtained from the collection of scaled DC coefficients in DCT compressed video is called a DC image. They are spatially reduced versions of the original images. Fig.3.2 shows an original image of size 352 × 288 and Fig.3.3 shows its DC image of size 44 × 36.

While the DC image is much smaller than the original image as it occupies only a small fraction of the original data size, it still retains most of the essential 'global' information. This suggests that scene change operations of a global nature that are performed on the original image can also be performed on the DC image. Operating on these DC images offers a significant saving in computations. In this section, it is shown how DC image can be effectively extracted from $MPEG$ compressed videos as suggested by Yeo and Liu [12].

Extraction of DC image from I-frame is trivial. The DC term $c(0,0)$ is related to the pixel values $f(i,j)$ by

$$c(0,0) = \frac{1}{8}\sum_{x=0}^{7}\sum_{y=0}^{7} f(x,y) \tag{3.5}$$

which is 8 times the average intensity of the block. Thus DC image is formed from blockwise averaging of the original image.

For P and B-frames, motion information must be employed to derive the DC image. To obtain the DC coefficients of the P-frame using the DCT coefficients from the pervious

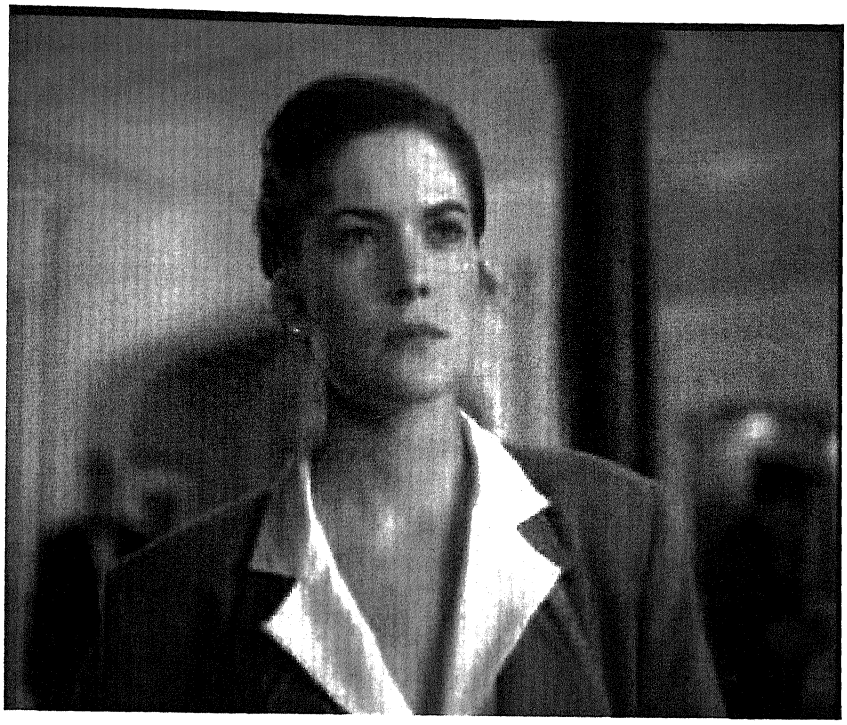


Figure 3.2: Full image at 352 × 288

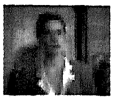


Figure 3.3: DC image at 44 × 36

I-frame, refer to Fig.3.4. P_{ref} is the current block of interest, $P_1, \ldots, P_4$ are the four original neighbouring blocks from which P_{ref} is derived. Let h_i and w_i be the height and width of $P_{ref} \cap P_i$. The shaded regions in $P_1, \ldots, P_4$ are moved by (w_1, h_1). Due to the linearity of the DCT, the DC coefficient of P_{ref} is of the form,

$$DC(P_{ref}) = \sum_{i=1}^{4} \left(\sum_{m=0}^{7} \sum_{l=0}^{7} w^i_{ml} \big(DCT(P_i) \big)_{ml} \right) \tag{3.6}$$

where we denote (m,l) component of P_i by $(P_i)_{ml}$. The factor w^i_{ml} weights the contribution of $(DCT(P_i))_{ml}$. The evaluation of eqn.3.6 may take a maximum of 256 multiplications. To reduce the number of multiplications Yeo[12] proposed first order approximation of eqn.3.6.

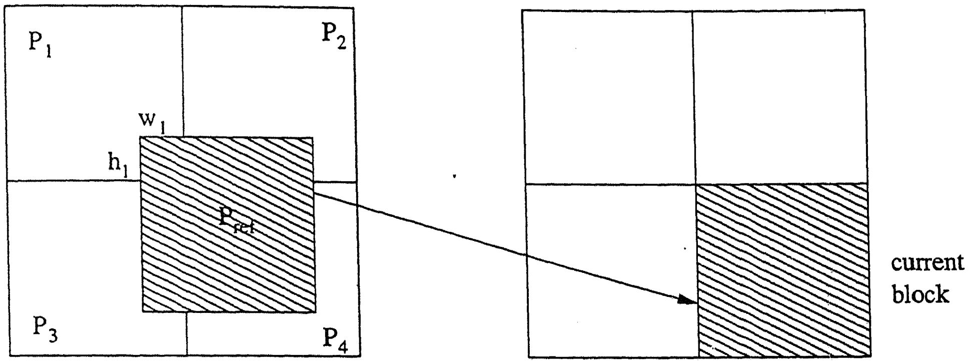


Figure 3.4: Reference Block(P_{ref}), Motion Vectors and Original Blocks

Under this approximation, the DC coefficient of P_{ref}, denoted as $DC(P_{ref})^1$, can be obtained by weighing the contributions from the 4 neighbouring DC values with the ratio of overlaps of the block P_{ref} with each of the blocks $P_1, \ldots, P_4$ respectively. That is,

$$DC(P_{ref})^1 = \sum_{i=1}^{4} \frac{h_i w_i}{64} DC(P_i) \tag{3.7}$$

Here at most 4 multiplications are required to obtain each DC value. Only DC coefficients of the 4 neighbouring blocks $\{P_i\}$, and motion vector information are used to obtain the DC values. Such information is easily obtained from MPEG compressed bit streams.

Under the first order approximation, DC value of P_{ref} given in eqn.3.6 can be written as a sum of first order approximation $DC(P_{ref})^1$, and an error term that does not depend on the DC coefficients of the reference blocks. i.e,

$$DC(P_{ref}) = DC(P_{ref})^1 + c \tag{3.8}$$

where c is the error due to first order approximation and is independent of the DC coefficients of P_i.

This approach can be extended to extract DC images of B-frames. It is observed from the expressions that extracting DC coefficients using the suggested approximation for P-frames yields DC images that are very close to the actual one.

3.5 Proposed Scene Change Detection Algorithm

Fernando *et al*[13] proposed an algorithm to detect gradual scene changes using statistical features (mean & variance) of luminance component of each frame. To detect abrupt scene changes they have used a different algorithm, which uses macroblock characteristics of the B-frames. In our proposed work we have modified this algorithm, such that both abrupt and gradual changes can be detected using the same data(mean & variance). We have also studied the effect of color on scene change detection.

3.5.1 Detection of abrupt scene change

We use a *sliding window* to examine the m successive frame differences. Frame differences are the difference between the mean values of the successive DC images. Let X_i, $i = 1, 2, 3, \ldots, N$ be a sequence of dc images. We form the difference sequence D_i, $i = 1, 2, 3, \ldots, N-1$ as

$$D_i = |d(X_i, X_{i+1})|. \tag{3.9}$$

Where $d()$ is the difference of the mean values of X_i and X_{i+1}. We declare a scene change from X_l to X_{l+1} if,

1. the difference is the maximum within a symmetric sliding window of size $2m-1$, i.e., $D_l \geq D_j, j = l-m+1, \ldots, l-1, l+1, \ldots, l+m-1$, and

2. D_l is also n times of the second largest maximum in the sliding window, where n should be at least 2.5. Performance of cut detection for different values of n is discussed in sec.6.2.1.

The parameter m is set to be smaller than the minimum duration between two scene changes. It is found experimentally in [14] that $m = 10$ gives good results.

Criterion 2 is imposed to guard against fast panning and zooming of scenes and also to prevent scenes with camera flashes to be declared as scene change. Fast panning and zooming usually manifest themselves as sequences of large values in the difference sequence, D_i. Flashes typically produce two consecutive sharp peaks. If D_l declares a scene change, then it is not necessary to check $D_{l+1}, \ldots, D_{l+m-1}$ and we can immediately proceed to check D_{l+m}.

The detection performance can be improved if we combine the results from both luminance and color. One-way of doing this is to declare a scene change from i to $i+1$ if there is a scene change in either the luminance or the color component. Besides improving detection performance it also increases the number of false detections. To avoid this problem we form the difference sequence D_i, $i = 1, 2, 3, \ldots, N-1$ as, the sum of differences of the mean values of luminance(Y) and color(C) of successive frames respectively, and the modified equation for D_i becomes,

$$D_i = |d_Y(X_i, X_{i+1})| + |d_C(X_i, X_{i+1})|. \tag{3.10}$$

Where d_Y and d_C are the differences of the mean values of the luminance and color components of X_i and X_{i+1} respectively.

This approach is found to perform better compared to the case of using only luminance component and the case of combining the results from both the components. The results of cut detection algorithm are reported in section.6.2.1.

3.5.2 Detection of Gradual scene changes

In video editing and production, two or more picture signals are simply added together so that the two pictures appear to merge on the output screen. Very often this process is used to move on from picture A to picture B. In this case, the proportions of the two signals are such that as the contribution of picture A changes from 100% to zero, the contribution of picture B changes from zero to 100%. This is called dissolving. When picture A is a solid color, the process is called fade-in, and when picture B is a solid color it is known as fade-out.

Mathematically dissolving can be expressed as follows:

$$S_n(x,y) = \begin{cases} f_n(x,y) & 0 \leq n < L_1 \\ \left[1 - \left(\frac{n-L_1}{F}\right)\right] f_n(x,y) + \left(\frac{n-L1}{F}\right) g_n(x,y) & L_1 \leq n \leq (L_1 + F) \\ g_n(x,y) & (L_1 + F) < n \leq L_2 \end{cases} \tag{3.11}$$

where $S_n(x,y)$ is the resultant video signal, $f_n(x,y)$ is picture A, $g_n(x,y)$ is picture B, L_1 is the length of sequence of picture A alone, F is the length of the dissolving sequence, and L_2 is the length of the total sequence.

It can be proven from Eqn.3.11 that during dissolving, the mean(m) and variance(σ) have a linear and quadratic behaviour, respectively, as shown in Eqn.3.12 and Eqn.3.13.

$$\begin{aligned} m_{s,n} &= E[S_n(x,y)] \\ &= \begin{cases} m_f & 0 \leq n < L_1 \\ \left[m_f - \frac{L_1}{F}(m_g - m_f)\right] - \frac{n}{F}(m_f - m_g) & L_1 \leq n \leq (L_1 + F) \\ m_g & (L_1 + F) < n \leq L_2 \end{cases} \end{aligned} \tag{3.12}$$

$$\begin{aligned} \sigma^2_{s,n} &= E[s_n^2] - E[S_n]^2 \\ &= \begin{cases} \sigma_f^2 & 0 \leq n < L_1 \\ \phi & L_1 \leq n \leq (L_1 + F) \\ \sigma_g^2 & (L_1 + F) < n \leq L_2 \end{cases} \end{aligned} \tag{3.13}$$

where

$$\phi = \xi n^2 - \left(\frac{2\sigma_f^2}{L} + 2L_1\xi\right)n + \left(\sigma_f^2 + L_1^2\xi + \frac{2L_1\sigma_f^2}{L}\right), \quad \xi = \frac{\sigma_f^2 + \sigma_g^2}{L^2} \tag{3.14}$$

Gradual transitions can be detected using the linearity & quadratic (parabolic) behaviour of mean & variance respectively. Fernando *et al* [13] proposed a method to combine these mean and variance to find gradual transitions. The ratio of first derivative of mean to second derivative of variance should be constant during a dissolve period, is taken as criterion for detecting gradual transitions in his method. In practice, this ratio is not a constant and it will have variations. In our proposed work, we suggest the following approach to detect gradual scene changes.

3.5.2.1 Detection of dissolves

As the variance has quadratic bahaviour in dissolve region, the differential values of variances of successive DC frames are negative for the first half of the dissolve region and positive for the next half of the dissolve region. The set of frames with these characteristics is $\{S\}$ and the number of frames in this set is F_1. If $F_1 \geq F$, where F is the number of frames in the dissolve region as given in Eqn.3.11, then there is a dissolve w.r.t variance, and it's presence is confirmed, if the mean values of $\{S\}$ have linear behaviour. If the differential values of means of successive DC frames of $\{S\}$ are all either positive or negative depending on whether the mean has positive or negative slope, then the presence of dissolve is confirmed and the starting frame of $\{S\}$ is taken as the starting frame of the dissolve.

3.5.2.2 Detection of fading

Both fade-in and fade-out can be considered as special cases of "dissolve" in which, one scene is a solid color. Fade-in starts with a solid color, and fade-out ends with a solid color. So the variance is zero at the start of fade-in and at the end of fade-out, as all the pixels have the same value, namely that of a solid color. Mean values of frames in fade region will exhibit linear behaviour. Thus, fade-in and fade-out are identified as follows:

Fade-in: Detect a frame with zero variance followed by a set of frames during which differential values of variances of successive DC frames are positive.

Fade-out: Detect a continuous set of frames during which differential values of variances of successive DC frames are negative followed by a frame with zero variance.

Experimental results of gradual scene change detection are reported in section 6.2.2.

Chapter 4

Keyframe Extraction and Clustering

In this chapter we discuss the methods used for *keyframe extraction* and *keyframe clustering*. *Keyframe extraction* involves the selection of representative frame(s) that represents the entire shot. *Keyframe clustering* refers to finding visual similarities between the keyframes of different shots of video. It reduces the redundancies in the keyframes of videos. Section 4.1 discusses keyframe extraction and in section 4.2 the proposed keyframe clustering algorithm is discussed.

4.1 Keyframe extraction

A video keyframe is the frame that can represent the salient content of a video shot. Keyframes provide a suitable abstraction for video indexing, browsing and retrieval. Users can quickly browse over the video by viewing only a few highlighted keyframes. The use of keyframes reduces the amount of data required in video indexing and browsing and provides an organizational framework for dealing with video content. It is important, that the choice of keyframes be made carefully, since a keyframe representing the entire shot will be needed for further processing.

The ideal method of selecting keyframes would be to compare each frame with one another in the scene and select the frame with the least difference from other frames in terms of a given similarity measure. But, it is a prohibitively expensive activity for large amount of video. Typical approaches to keyframe selection involve choosing a frame from a fixed position in the scene, or several frames separated by fixed distance. Choosing a frame in a fixed position in a scene (e.g. first, middle, last) can miss the content and action present in a scene. Better representative keyframes could be chosen if scene content were considered in the selection. "Scene content" involves pixel values and similar global measures of a scene.

The algorithm we used for keyframe extraction uses kurtosis of the pixel values in a frame to select the representative keyframes[15]. Kurtosis, also called the excess coefficient, measures the degree of peakedness of a distribution, and it is expected to be better suited for selection of keyframes since it gives more weight to the shape of the distribution. Kurtosis is denoted as γ_2 (or b_2) and is computed by taking the fourth moment of a distribution. Let μ_i denote the i-th moment. Then the Fisher kurtosis used in our calculations is defined by

$$\gamma_2 = b_2 = \frac{\mu_4}{\mu_2^2} - 3 = \frac{\mu_4}{\sigma^4} - 3 \tag{4.1}$$

Kurtosis for all the frames in a video sequence are calculated. The frames with the highest and lowest kurtosis are the candidates for the representative keyframes. It is found in[15] that kurtosis based selection of keyframes perform better than the algorithms using average frame as well as histogram difference metrics for selecting keyframes. In the first group of algorithms, an average frame for the whole scene is calculated and the frame whose distance is maximum or minimum from the average frame is selected as the candidate keyframe. Whereas in the second group of algorithms histogram difference between the successive frames is calculated and the frame with maximum or minimum distance is the candidate keyframe. The algorithm that uses kurtosis is found to perform better than the above mentioned algorithms.

4.2 Keyframe clustering

Visual similarity is usually present in different video shots due to same locations or persons or events. The need to find the shots of similar contents is thus of considerable interest. In addition, underlying story structure inherent in a video document is often reflected by repeating scenes of similar visual contents. By grouping video shots together, a compact representation of video can be obtained. This grouping of shots can be achieved by comparing the keyframes of the shots, since every shot is represented by a keyframe. Finding visual similarities between the keyframes of different shots is keyframe clustering. Besides producing a compact representation of video, keyframe clustering also reduces the retrieval time as the size of keyframe database with keyframe clustering is less than the size of keyframe database without clustering.

4.2.1 Method of Clustering

Keyframe clustering starts with the second keyframe of every video. Every keyframe starting from the second keyframe is compared with the clustered keyframes of that video. For each keyframe best match is found from the clustered keyframes. If the similarity distance between the best match and the keyframe is less than a threshold(ϵ) then this keyframe is clustered with that best match. If the distance is greater than ϵ then there is no match to the keyframe, and is added to the list of clustered keyframes. For similarity distance calculations between keyframes, color layout descriptor [3] is used. As explained later, similarity distance function between keyframes is the difference between DCT coefficients of the keyframes. If the two keyframes that are being compared have similar spatial distribution, then the frequency distribution will be similar and the difference between DCT coefficients will be small. So, the value of ϵ should be small. A typical value of $\epsilon = 1$ is chosen. Performance of keyframe clustering for different values of ϵ is discussed in sec.6.3.

4.2.2 Color Layout Descriptor method

Color layout descriptor(CLD) specifies the spatial distribution of colors in keyframes. The extraction of descriptor consists of four stages: keyframe partitioning, dominant color extraction, DCT transform, zigzag scanning of DCT coefficients. This is illustrated in Fig.4.1. In the first stage, every keyframe is partitioned into 64 blocks. The size of each block is $W/8 \times H/8$ where W and H denote the width and height of an input picture, respectively. In the second stage, a single dominant color is selected in each block to build a tiny image whose size is 8×8. Any method for dominant color selection can be applied. We use simple average colors as the dominant colors. In the third stage, both the components (Y and C) are transformed by 8×8 DCT, and we obtain 2 sets of DCT coefficients. A few low frequency coefficients are extracted using zigzag scanning. It is found in [3] that optimized descriptor requires 6 coefficients for luminance and 3 coefficients for color components. In the present approach we use 6 coefficients for Y and 3 coefficients for C to represent each keyframe. Two out of these 9 coefficients are DC coefficients and other 7 coefficients are AC coefficients. DC coefficients are the zero frequency coefficients of the DCT transformed luminance(Y) and color(C). AC coefficients are the coefficients other than DC coefficients of the DCT transformed luminance(Y) and color(C).

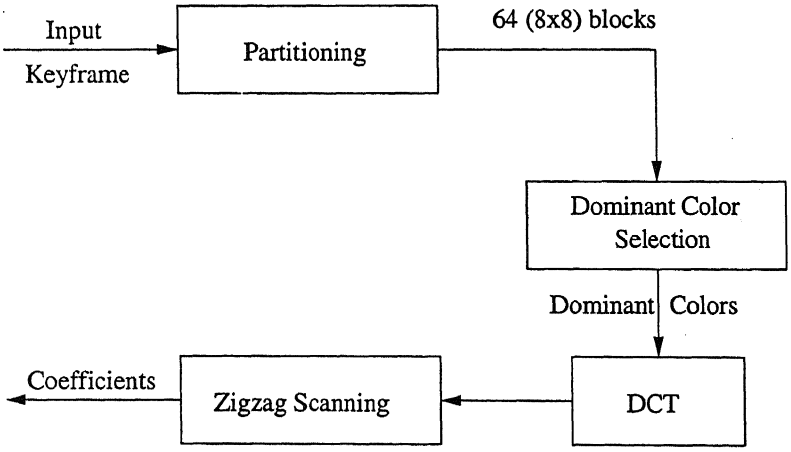


Figure 4.1: Block diagram of the color layout descriptor extraction

4.2.2.1 Similarity distance function

To find similarity between two different keyframes, color layout descriptor coefficients of the keyframes are compared. Similarity distance between two color layout descriptors is expressed in equation form as follows:

$$D = \sqrt{\sum_{i=1}^{6} w1_i(Y1_i - Y2_i)^2} + \sqrt{\sum_{i=7}^{9} w2_i(C1_i - C2_i)^2} \qquad (4.2)$$

where D is distance, and $w1_i$ and $w2_i$ are the weighing values for the i-th coefficient. Y1 & Y2 are CLD coefficients of luminance, and C1 & C2 are CLD coefficients of color(C) of the two keyframes which are compared.

Using this similarity measure keyframe clustering is performed as described in sec.4.2.1. Results of keyframe clustering are reported in sec.6.3.

Chapter 5

Indexing and Retrieval

Keyframes form the basis for indexing each video partition(shot). Further applications such as, video browsing, and content based retrieval of video, depends on the efficient indexing of these keyframes. A straight forward approach of indexing these keyframes is to represent the visual contents in textual form(e.g. keywords and attributes). But, there are several problems with this method. First of all, human intervention is required to describe and tag the contents of the visual data in terms of the selected set of captions and keywords. In video, there are several objects that could be referenced, each having its own set of attributes. As the size of the database grows, the use of keywords not only becomes complex but also inadequate to represent the video content. If the video database is to be shared globally, then the linguistic barriers will make the use of keywords ineffective. To overcome these difficulties, content-based video retrieval emerged as a promising means for describing and retrieving videos. Content-based video retrieval systems describe videos by their own visual content rather than text, such as color and texture. Since keyframes are the representative frames of the video, they are indexed by one or a combination of the above features.

In the following section a brief introduction to content descriptors of video is given.

5.1 Video Content Descriptors

Video content can be described by visual features like color, texture; and motion features of shots. Video content can be described by one or a combination of these features.

5.1.1 Visual Color Descriptors

Color is one of the most widely used visual features in image and video retrieval. Color features are relatively robust to changes in the background colors and are independent of image size and orientation. Color descriptors can be used for describing content in still images and video, respectively.

Scalable Color Descriptor (SCD) : One of the most basic descriptions of color features is provided by describing color distribution in images. If such a distribution is measured over an entire image, global color features can be described. The MPEG-7 generic SCD is a color histogram encoded by a Haar transform. It uses the HSV color space uniformly quantized to 255 bins. To arrive at a compact representation the histogram bin values are nonuniformly quantized in a range from 16 bits/histogram for a rough representation of color distribution and up to 1000 bits/histogram for high-quality applications. Matching between SCD realizations can be performed by matching the Haar coefficients or histogram bin values employing an L1 norm.

Dominant Color Descriptor : This color descriptor aims to describe global as well as local spatial color distribution in images for high-speed retrieval and browsing. In contrast to the color histogram approach, this descriptor arrives at a much more compact representation at the expense of lower performance in some applications. Colors in a given region are clustered into a small number of representative colors. The descriptor consists of the representative colors, their percentages in a region, spatial coherency of the color, and color variance.

Color Structure Descriptor(CSD) : The main purpose of the CSD is to express local color features in images. To this aim, a pel structuring block scans the image in a sliding window approach. With each shift of the structuring element, the number of times a particular color is contained in the structure element is counted, and a color histogram is constructed in such a way.

Group-of-Frames/Group-of-Pictures (GoF/GoP) Color Descriptor : The *GoF/GoP* color descriptor defines a structure required for representing color features of a collection of similar frames or video frames by means of the SCD. It is useful for retrieval in image and video databases, video shot grouping, image-to-segment matching, and similar applications. It consists of average, median, and intersection histograms of groups of frames calculated based on the individual frame histograms.

5.1.2 Visual Texture Descriptors

Texture refers to the visual patterns that have properties of homogeneity or not, that result from the presence of multiple colors or intensities in the image. It is a property of virtually any surface, including clouds, trees, bricks, hair and fabric. Describing textures in images by appropriate texture descriptors provides powerful means of similarity matching and retrieval. Some descriptors of texture are *homogenous texture descriptor*, and *non-homogenous texture descriptor*[5].

5.1.3 Motion Descriptors for Video

Description of motion features in video sequences provides information about its content. In general, describing motion in video by motion fields can be very expensive in terms of bits per image, even if motion vector fields are coarse. MPEG-7 has developed descriptors that capture essential motion characteristics from the motion field into concise and effective descriptions. *Motion activity descriptor, camera motion descriptor* and *Motion trajectory descriptor* are some of the MPEG-7 motion descriptors[5].

5.2 Indexing Keyframes

In this section the method of indexing keyframes is discussed. We use color layout descriptor proposed in[3] to index keyframes. Color Layout Descriptor extraction is explained in section 4.2.2 in the context of keyframe clustering. As explained in that section, we use 6 coefficients for Y and 3 coefficients for C to index each keyframe.

For the purpose of comparison with $3D$ representation of color, indices are extracted both in YC domain and $YCbCr$ domain. Color Layout Descriptor algorithm[3] is proposed for $YCbCr$. Hence $3D$–$YCbCr$ representation is chosen for comparison with $2D$–YC. For $YCbCr$ representation, 6 coefficients for Y, 3 coefficients for Cb, and 3 coefficients for Cr are taken as given in [3]. These indices are stored in the database along with the keyframes to facilitate the retrieval of video segments. The advantage of indexing in YC domain is that we can represent each keyframe with only 9 coefficients, while YCbCr requires 12 coefficients. This reduces the storage cost in storing the indices. The reduction in storage in terms of number of bits is discussed in sec.6.4

5.2.1 Advantages of color layout descriptor

The advantages of this descriptor are:

- that there is no dependency on image/video format, resolutions, and bit-depths. The descriptor can be applied to any still picture or video frames even though their resolutions are different. It can be also applied both to a whole image and to any connected or unconnected parts of an image with arbitrary shapes.

- that the required hardware/software resources for the descriptor are very small. It needs as low as 6 bytes(YC) per image in the default video frame search, and the calculation complexity of both extraction and matching is very low. It is feasible to apply this descriptor to mobile terminal applications where the available resources is strictly limited due to hardware constrain.

- that the captured feature is represented in frequency domain, so that users can easily introduce perceptual sensitivity of human vision system for similarity calculation.
- that it supports scalable representation of the feature by controlling the number of coefficients enclosed in the descriptor. The user can choose any representation granularity depending on their objectives without interoperability problems in measuring the similarity among the descriptors with different granularity. The default number of coefficients is 9(YC).

5.3 Retrieval

Retrieval is one of the main applications of video indexing. A content-based video retrieval system is a querying system that uses content as a key for the retrieval process. Since keyframes represent the content of the video, whenever a query is posed, it is compared with the keyframes that are stored in the database. A ranked set of keyframes with high matching scores is presented. The retrieval of video sequence follows two steps:
step1: The query is processed and relevant features are extracted.
step2: The features are compared with the features of the keyframes stored in the database, and best matches are retrieved.

We implemented two types of querying mechanisms: *query by frame* and *query by clip.*
Query by frame: The query frame is processed and indices of the query are extracted as explained in sec.5.2. These indices are compared with the indices of the keyframes in the database. The keyframes and associated video segments which have least distance from the query are retrieved.
Query by clip: In this case the query is a clip. Query clip is processed to detect scene changes, and to extract keyframes. For each keyframe the matched segments are retrieved as described in *query by frame.*

5.3.1 Similarity/Distance measures

When a query is posed to the system, indices of the query are extracted and they are compared to the indices stored in the database. Distance between two descriptors is calculated as follows:

For YC representation:

$$D = \sqrt{\sum_{i\in(Y)} w1_i(Y_i - Y_i^1)^2} + \sqrt{\sum_{i\in(C)} w2_i(C_i - C_i^1)^2} \tag{5.1}$$

Here, Y_i, C_i denote the i-th coefficient of Y,C color component and $w1_i$ and $w2_i$ are the weighing values for the i-th coefficient respectively. The weighing values should decrease according to the zigzag-scan order. The recommended values[3] are power of 2 to accelerate the speed with only shift operations.

For $YCbCr$ representation:

$$D = \sqrt{\sum_{i\in(Y)} w1_i(Y_i - Y_i^1)^2} + \sqrt{\sum_{i\in(Cb)} w2_i(Cb_i - Cb_i^1)^2} + \sqrt{\sum_{i\in(Cr)} w3_i(Cr_i - Cr_i^1)^2} \tag{5.2}$$

Here, Y_i, Cb_i and Cr_i denote the i-th coefficient of Y,Cb,Cr color component, and $w1_i$, $w2_i$ and $w3_i$ are the weighing values for the i-th coefficient respectively.

The similarity calculation cost is proportional to the number of coefficients enclosed in the descriptor. From Eqn.5.1 and Eqn.5.2 we can say that, distance calculation cost in YC domain is less than that of YCbCr. The reduction in retrieval time is discussed in sec.6.5.

Chapter 6

Experimental Results

In this chapter we present the simulation results obtained for all the experiments conducted on various video sequences.

6.1 Results on YC Transformed Images

To evaluate the performance of spiral mapping we experimented on different images containing a variety of colors, each of size 256 × 256 in the RGB color space. The RGB values of the pixels are transformed to Y,UU,VV values. The Y value is preserved, and UU,VV values are approximated on to a spiral for a particular number of encirclements L to get the color value C. All the pixel values of an image are now represented by two quantities Y and C, instead of the three quantities R, G and B. To recover original image from YC representation, an inverse operation is performed. Using C, the approximated values for UU and VV are obtained. The Y, UU and VV are transformed back to RGB. This is done for all the pixels in an image.

6.1.1 Experimental results for varying L and corresponding $PSNR$

Spiral transformation was carried out for different test images, and for different number of encirclements L. Original and reconstructed images for different L values are shown in Figs.6.1–6.2. A numerical evaluation of the reconstructed image is made by computing the $PSNR$ as given in Eqn.6.1 between the original(O) and the reconstructed(R) images respectively.

$$PSNR = 10log_{10}\left[\frac{3 * 255^2 * N^2}{\sum_{i=1}^{3}\sum_{j=1}^{N}\sum_{k=1}^{N}\left[O(i,j,k) - R(i,j,k)\right]^2}\right] \tag{6.1}$$

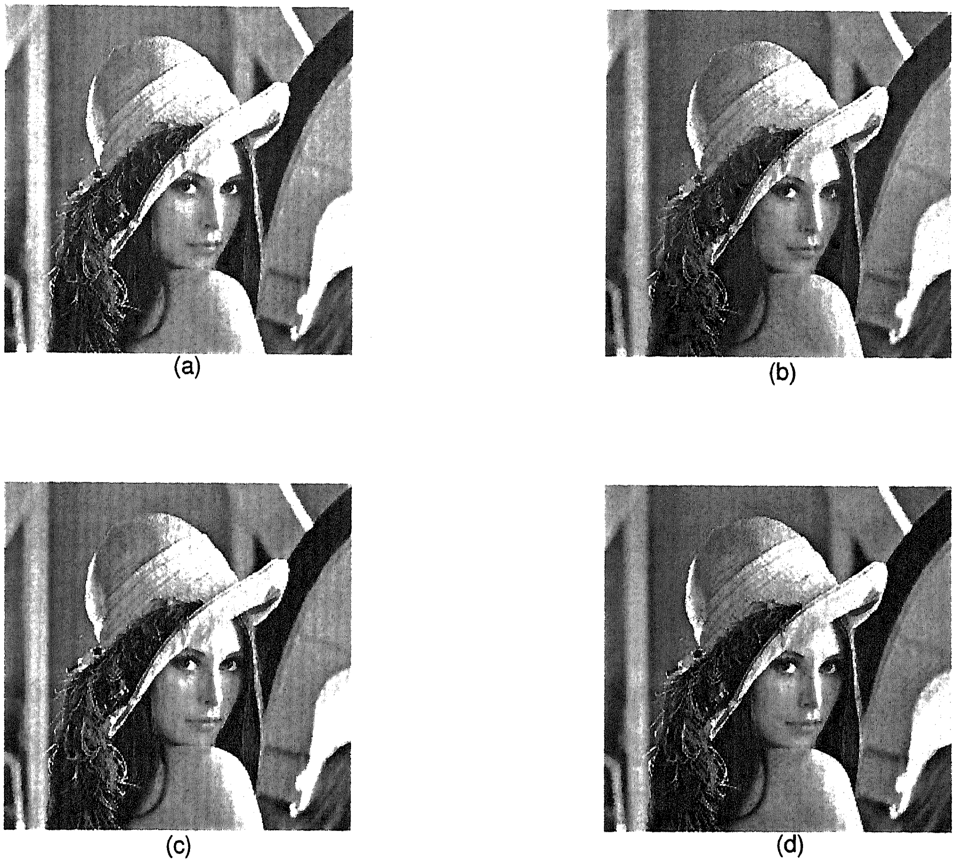


Figure 6.1: Reconstructed lena Images For Different Values of L, (a) original image, (b) L=3, (c) L=5, (d) L=7

Table.6.1 shows the $PSNR$ values of some of the test images for different values of L. The color loss in processed images is indistinguishable. By observing the processed images for different values of L, we note that $L = 7$ yields the best results.

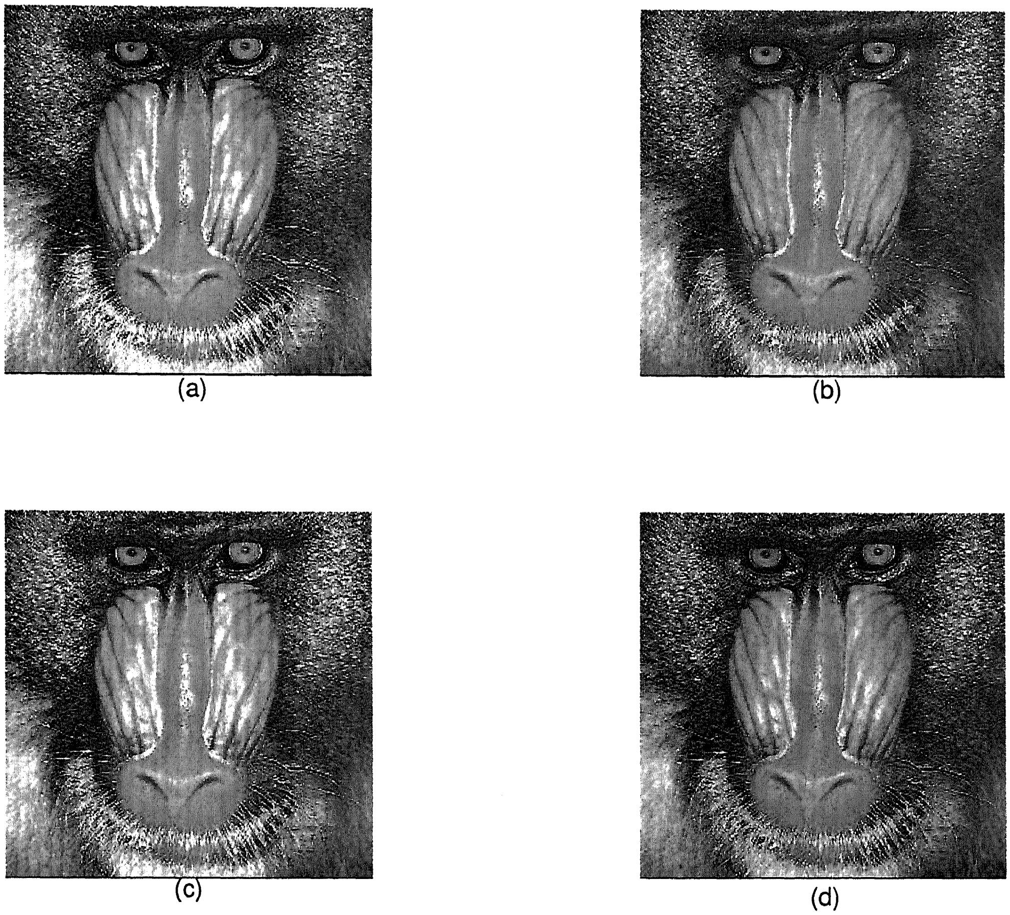


Figure 6.2: Reconstructed baboon Images For Different Values of L, (a) original image, (b) L=3, (c) L=5, (d) L=7

S.No.	Image	PSNR for (l=3)	PSNR for (L=5)	PSNR for (L=7)
1	lena	26.8129	33.3561	34.1870
2	baboon	25.4863	30.5228	33.9277
3	flowers	24.7301	29.6687	33.0210
4	snow	22.4478	28.0073	31.9777

Table 6.1: PSNR values for reconstructed images

6.2 Performance of scene change detection algorithm

To evaluate the performance of the proposed method, we experimented on different videos of varying characteristics. DC images of the compressed videos are extracted as discussed in sec.3.4. All these DC images are mapped from RGB to YC domain. Mean and variance of Y and C of all the DC images are calculated.

6.2.1 Performance of cut detection

For the detection of cuts differential means of the successive DC images are calculated. Fig.6.3 shows the plot of differential means(D_i) of the DC images of a movie clip from the movie Baby's day out. The movie clip has 4104 frames and 30 abrupt scene changes(cuts). The figure shows the plots for both the cases of cut detection. The first plot is for the case when luminance(Y) means are used and the second plot is when luminance(Y) and color(C) means are used. From the figure we observe that, there are miss detections at frame numbers 3255 and 3797 when only Y is used, while they are detected when both Y and C are used. Similar observation can be made from Fig.6.4 where there is a miss detection at frame number 2302 when only Y is used, while it is detected when both Y and C are used. There is a miss detection at frame number 2344 when both Y and C are used, while it is detected when only Y is used. For the values of $m = 10, \ n = 3 \ and \ L = 7$, there is 1 miss detection and 4 false detections when both Y and C are used, while there are 3 miss detections and 3 false detections when only Y is used. The figures 6.3 and 6.4 show the advantage of using both luminance and color versus using only luminance, in cut detection.

6.2.1.1 Performance of cut detection for different values of n

Lower values of n has the advantage of less number of miss detections, and the disadvantage of large number of false detections. While the reverse happens for higher values of n, i.e, higher number of miss detections and less false detections. So, the value of n should be selected such that the number of miss detections and false detections is moderate. There

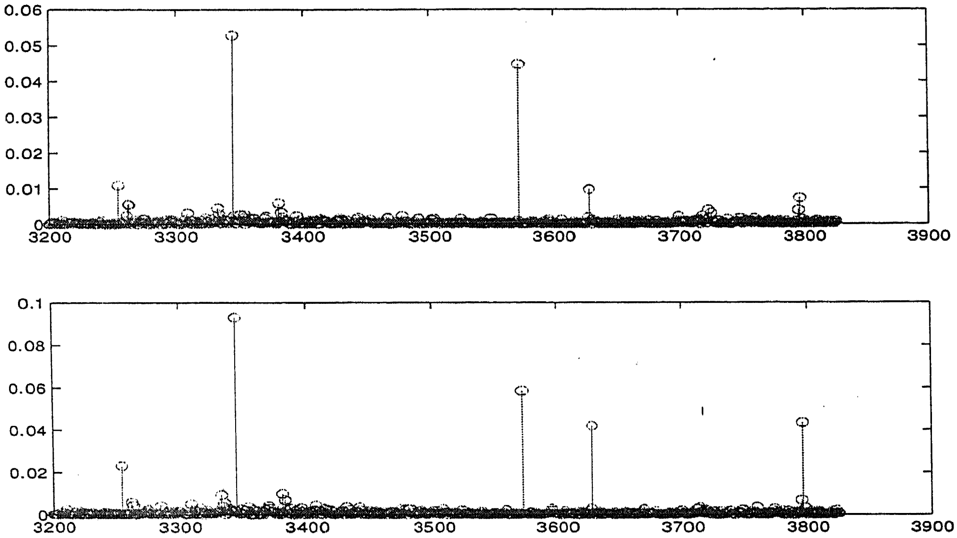


Figure 6.3: Plot of D_i *vs* frame number(i), i:3200:3900

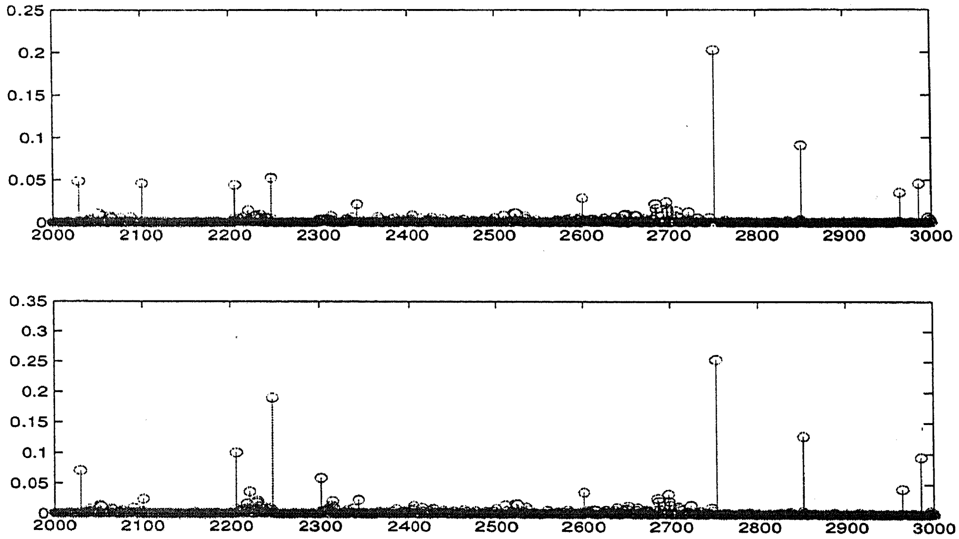


Figure 6.4: Plot of D_i *vs* frame number(i), i:2000:3000

are two parameters Precision and Recall that are used to evaluate the performance of scene change detection algorithm. They are defined as follows:

$$\begin{aligned} Recall(R) &= \frac{N_t - N_d}{N_t} \\ Precision(P) &= \frac{(N_t - N_d)}{((N_t - Nd) + N_i)} \end{aligned} \tag{6.2}$$

where,

N_i : Number of false shot boundary detected by the method(false detections),

N_d : Number of shot boundary not detected by the method(miss detections), and

N_t : Number of actual shot boundary in the baseline.

The value of n should be selected to have a high value of both precision and recall. Table 6.2 gives the number of miss detections and false detections of a movie clip from the movie Baby's day out, for different values of n, and for $m = 10$ *and* $L = 7$. From the table it is clear that the values of n ranging from 2.75–3.75 give good results.

n	2.0	2.25	2.5	2.75	3.0	3.25	3.5	3.75	4.0
N_d	0	0	0	1	1	2	2	2	3
N_i	26	17	9	6	4	1	1	1	0
R	100	100	100	96.66	96.66	93.33	93.33	93.33	90
P	53.57	63.82	76.92	82.85	87.87	96.55	96.55	96.55	100

Table 6.2: Performance of cut detection for different n and m=10,L=7(Both Y&C)

Table.6.3 illustrates the performance of cut detection for the same video when only luminance(Y) is used to detect cuts. From Tables 6.2 and 6.3 we can infer that for same values of n, the scene change detection algorithm performs better when both Y and C are used compared to using only Y.

Table.6.4 summarizes the results of cut detection for different videos, for the case when $n = 3, m = 10,$ *and* $L = 7$. 'Bdout3.mpg', 'Bdout9.mpg' are movie clips from the movie Baby's day out. 'FN1' is movie clip from the animation movie Finding Nemo. 'Martin' is a music video of Ricky Martin. These four videos have different characteristics. Music videos

n	2.5	2.75	3.0	3.25	3.5
N_d	3	3	3	3	3
N_i	12	3	3	2	2
R	90	90	90	90	90
P	69.23	81.81	90	93.10	93.10

Table 6.3: Performance of cut detection for different n and m=10,L=7(Only Y)

generally contain fast motion sequences, and video content changes very rapidly. Movies are generally of slow motion, and content changes are relatively less compared to music videos. Animation movies have different scene content compared to movies. Hence, these videos are taken for evaluation of cut detection algorithm.

Video	Total cuts	Miss detections	False detections	Recall	Precision
Bdout3.mpg	48	3	3	93.75%	93.75%
Bdout9.mpg	30	1	4	96.67%	87.88%
FN1	25	4	3	84.00%	87.50%
Martin	66	4	9	93.94%	87.32%

Table 6.4: Performance of cut detection for different videos

6.2.2 Performance of gradual scene change detection

Two test videos are used to evaluate the performance of the gradual scene change detection algorithm. The first video has 4087 frames and 4 dissolves. The second video has 4952 frames and 6 fades: 3 fade-in, 3 fade-out. First and second videos are used to evaluate the performance of dissolve and fade detection respectively.

6.2.2.1 Results of dissolve detection

The first two plots in Fig.6.5 show the variation of mean and variance respectively from frame number 2740 to frame number 2780 of the first video. There is a dissolve starting at frame number 2753. The linear and quadratic behaviour of mean and variance respectively can be clearly seen in the figure. The third and fourth plots of the figure show the plots of differential values of mean and variance respectively.

The value of F is taken as 10, where F is number of frames in the dissolve region given in eqn.3.11. So it detects dissolves and fades of duration greater than or equal to 10 frames. To detect dissolves, at first differential variances of DC frames are checked and, if there is any pattern of dissolve, then it's presence is confirmed by checking the means of corresponding frames. Fig.6.6 shows the variation of mean and variance respectively of another dissolve which starts at frame number 107.

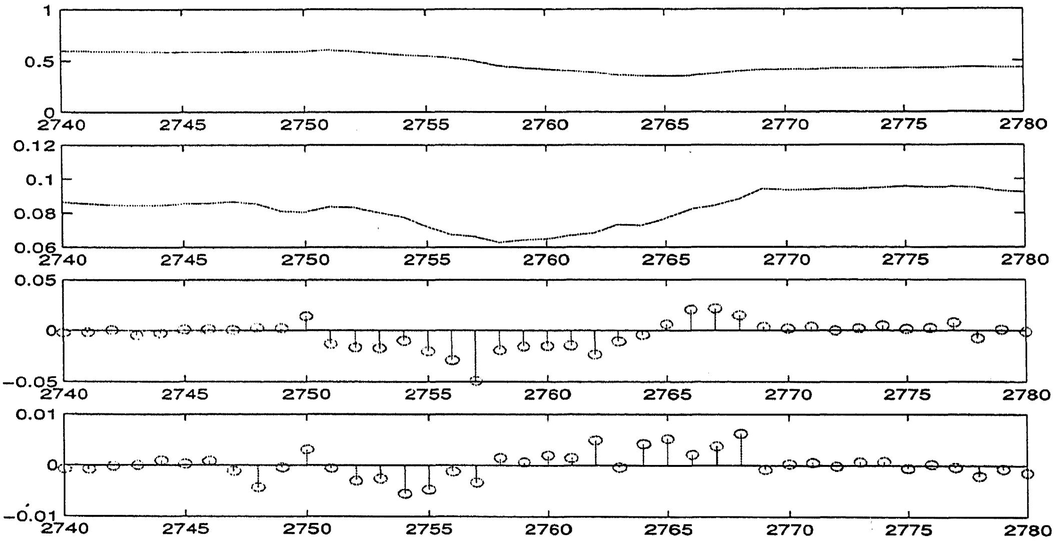


Figure 6.5: From top to bottom: Plots of mean, variance, differential mean, differential variance *vs* frame number

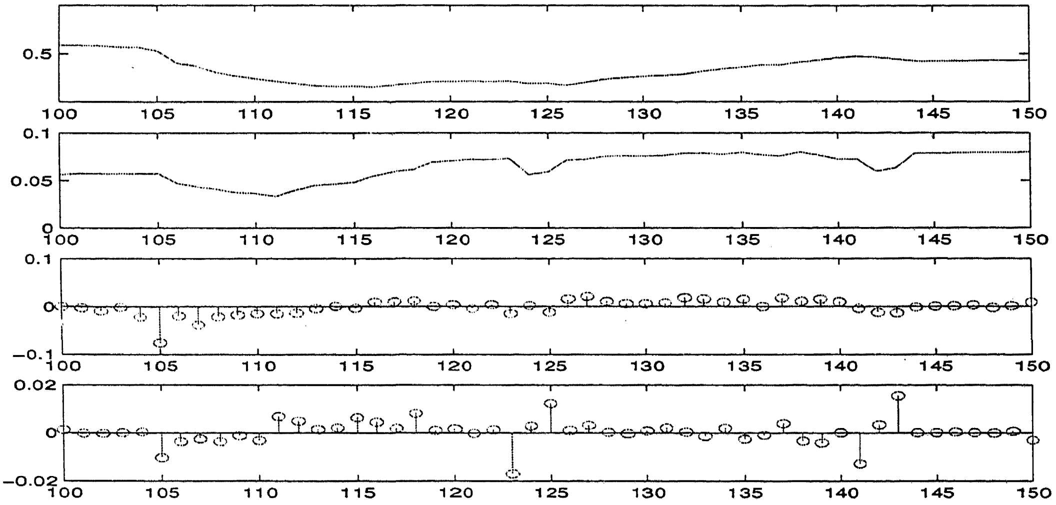


Figure 6.6: From top to bottom: Plots of mean, variance, differential mean, differential variance *vs* frame number

6.2.2.2 Results of fade detection

Fig.6.7 shows the behaviour of mean and variance respectively, from frame number 2100 to frame number 2200. There is a fade-out which starts at frame number 2107 and a fade-in region which starts at frame number 2126. From the figure it is clear that mean and variance have linear and quadratic properties respectively, and fade-out ends with a solid color frame with zero variance. Similarly fade-in starts with a solid color frame with zero variance. The same value of $F = 10$ is used to find the fades. Fig.6.8 shows the variations of mean and variance of another fade-out, fade-in pair which starts at frame numbers 3310 and 3328 respectively.

6.3 Performance evaluation of keyframe clustering

In this section results of keyframe clustering algorithm are presented. The proposed keyframe clustering algorithm uses color layout descriptor for finding similarities between the keyframes

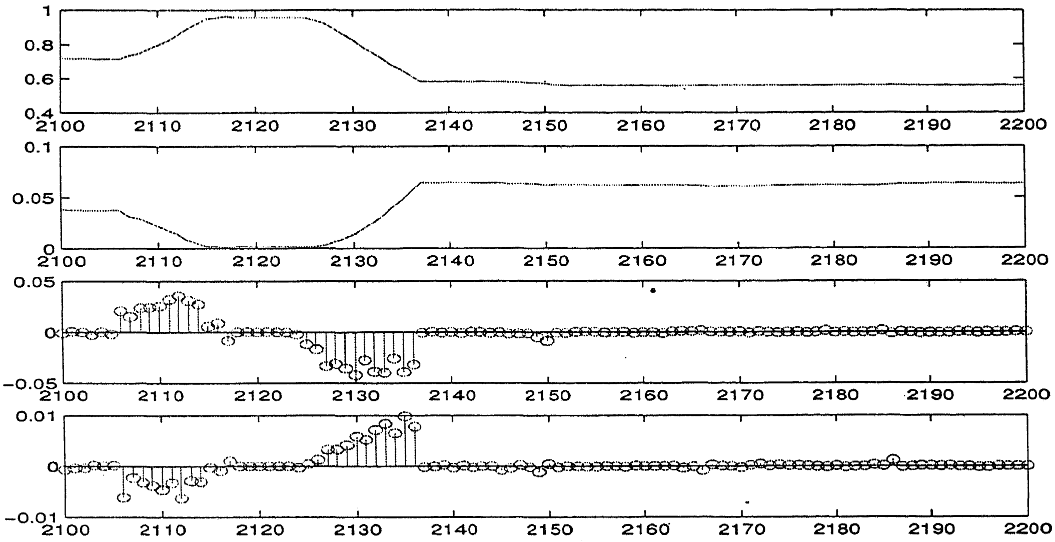


Figure 6.7: From top to bottom: Plots of mean, variance, differential mean, differential variance *vs* frame number

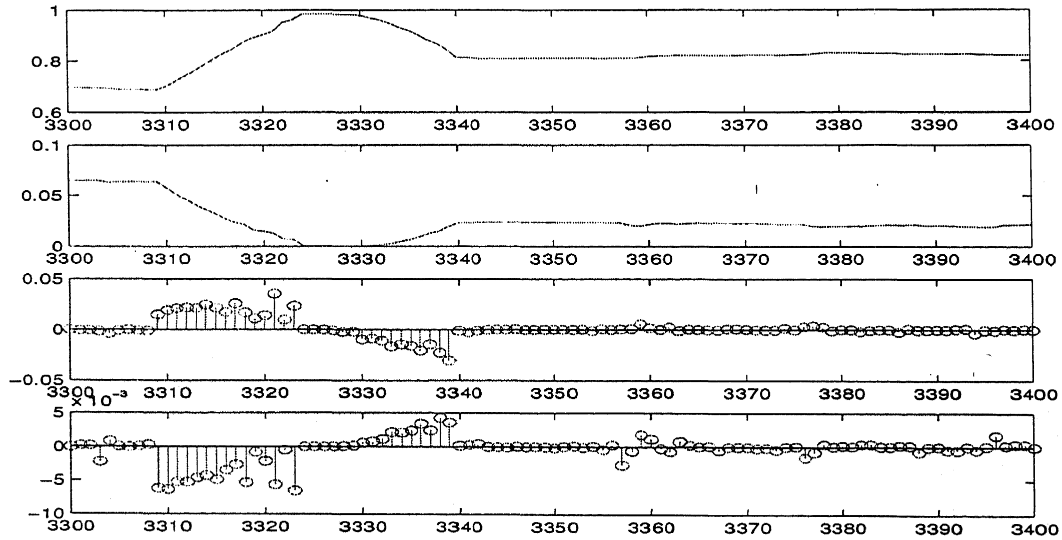


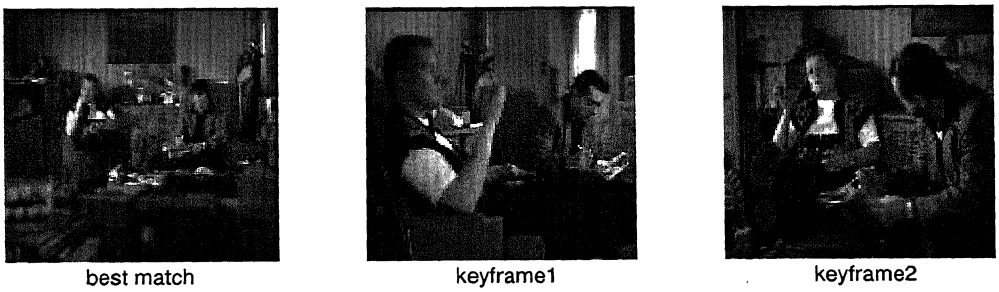
Figure 6.8: From top to bottom: Plots of mean, variance, differential mean, differential variance *vs* frame number

of different shots. Color layout descriptor coefficients for all the keyframes are extracted as described in sec.4.2.2. Keyframe clustering is performed as described in sec.4.2.1. Every keyframe of the video is compared to the clustered keyframes and the best match is found. If the distance between the best match and the keyframe as given in eqn.4.2 is less than a threshold(ϵ) then this keyframe is clustered with that best match. Table.6.5 shows the performance of keyframe clustering for different thresholds(ϵ),where C represents the number of keyframes that are clustered correctly, while F represents the number of keyframes that are falsely clustered. These results are for a movie clip from the movie Baby's day out which has 49 keyframes. There are 26 keyframes which have visual similarity with other keyframes.

Threshold(ϵ)	0.75	1.0	1.25	1.5	1.75
C	19	21	23	25	26
F	0	0	0	1	2

Table 6.5: Performance for different thresholds(ϵ)

Figures 6.9–6.11 show three different examples of keyframe clustering. In these figures all the frames are keyframes of different shots. The keyframes labeled 'keyframe1' and 'keyframe2' in each figure are clustered to the frame with label 'best match' in the same figure. Now the shots corresponding to 'keyframe1' and 'keyframe2' can be represented with 'best match'.

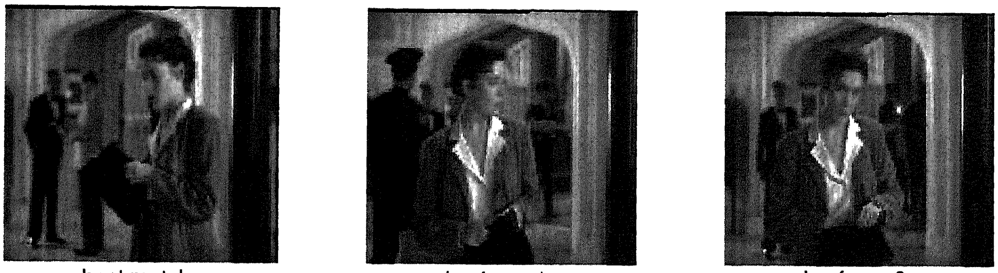


best match

keyframe1

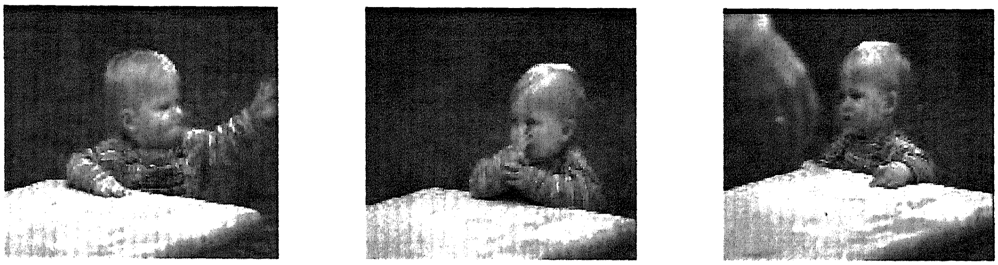
keyframe2

Figure 6.9: Example of keyframe clustering: 1



best match keyframe1 keyframe2

Figure 6.10: Example of keyframe clustering: 2



best match keyframe1 keyframe2

Figure 6.11: Example of keyframe clustering: 3

Table.6.6 illustrates the performance for different videos. Bdout3.mpg, Bdout4.mpg, Bdout9.mpg are clips from the movie baby's day out. These results are for $\epsilon = 1.5$. The average detection performance is better compared to the detection performance of 69.7% obtained in[16] while false detections are less than 8.86%.

	Detections	Misses	False detections
Bdout3.mpg	96.15%	3.86%	3.84%
Bdout4.mpg	83.33%	16.67%	4.08%
Bdout9.mpg	75.00%	25.00%	16.67%
average	84.83%	15.17%	8.20%

Table 6.6: Detection performance of keyframe clustering

6.4 Indexing

Indexing follows keyframe clustering. Indices of clustered keyframes of the video are extracted as described in sec.5.2. Indices of the keyframes are extracted both in YC domain and YCbCr domain, for comparing the performance of YC representation with 3D color representation. YC representation needs 9 coefficients to represent each keyframe, 6 for luminance(L) and 3 for color(C). Two of these 9 coefficients are DC coefficients, and the remaining 7 are AC coefficients. YCbCr representation requires 12 coefficients to represent each keyframe: 6 for Y, 3 for Cb, 3 for Cr. Among these 12 coefficients, 3 are DC coefficients and the remaining are AC coefficients.

If the descriptor coefficients are coded with 6 bits for DC coefficients and 5 bits for AC coefficients [3], then each keyframe descriptor in YCbCr domain takes 63 bits to store, while each keyframe descriptor in YC domain takes only 47 bits. There is a reduction of 16 bits per descriptor, if keyframes are indexed in YC domain. This shows that storage cost in indexing videos in YC domain is significantly less compared to indexing videos in 3D YCbCr domain.

6.5 Retrieval

When a query is given to the system the indices of the query are extracted. These indices are compared to the indices of the keyframes stored in the database. Distance of the query from all the keyframes in the database is calculated. Those keyframes with least distance from the query and the corresponding video segments are retrieved.

6.5.1 Comparison of retrieval performance of YC & YCbCr

Fig.6.12–6.19 shows the results of retrieval in both YC and YCbCr domain. In all these figures the frame with title 'query' is the query and other frames with titles 'Rank' are the retrievals with corresponding ranks. In all these results Query frame is retrieved in the first position, and other relevant frames are also retrieved. Retrieval results in YC domain are comparable to that of YCbCr domain. The advantage of video retrieval in YC domain is that retrieval time of YC is less than that of YCbCr. For a database containing 936 keyframes, retrieval time in YCbCr domain is 0.0982 seconds, while retrieval time in YC domain is 0.0840 seconds. This amounts to a reduction of 14.46% in retrieval time. The difference between the retrieval times of YC and YCbCr increases as the size of database increases. These results demonstrate the advantage of video indexing and retrieval in two-dimensional YC domain.

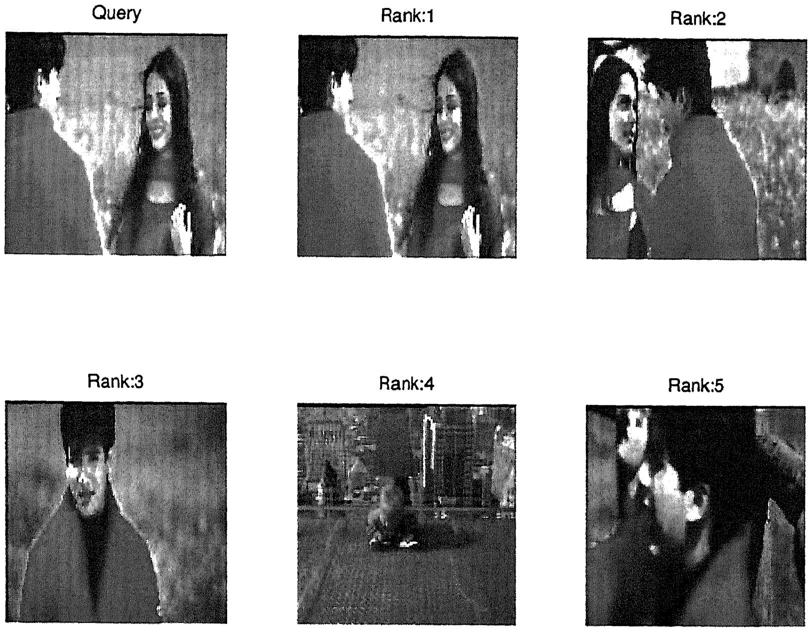


Figure 6.12: Query1:Retrieval in YC domain

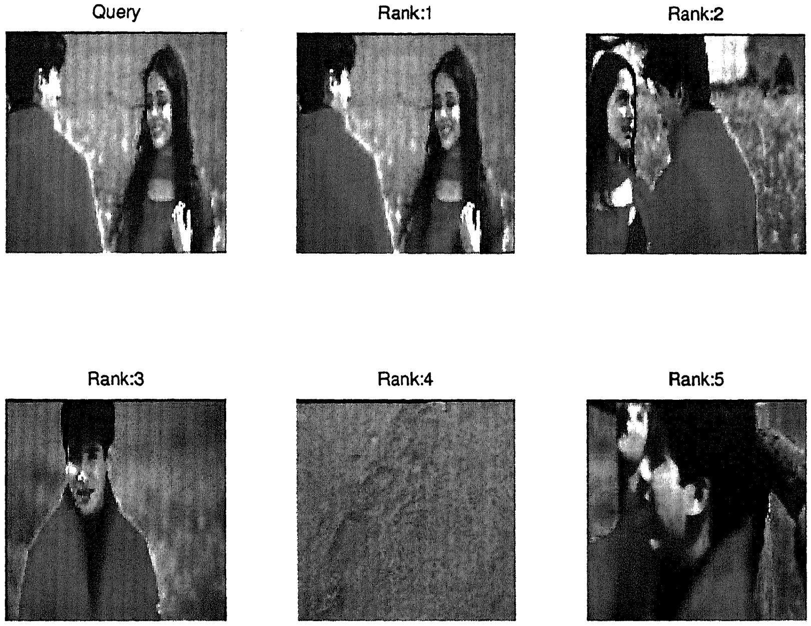


Figure 6.13: Query1:Retrieval in YCbCr domain

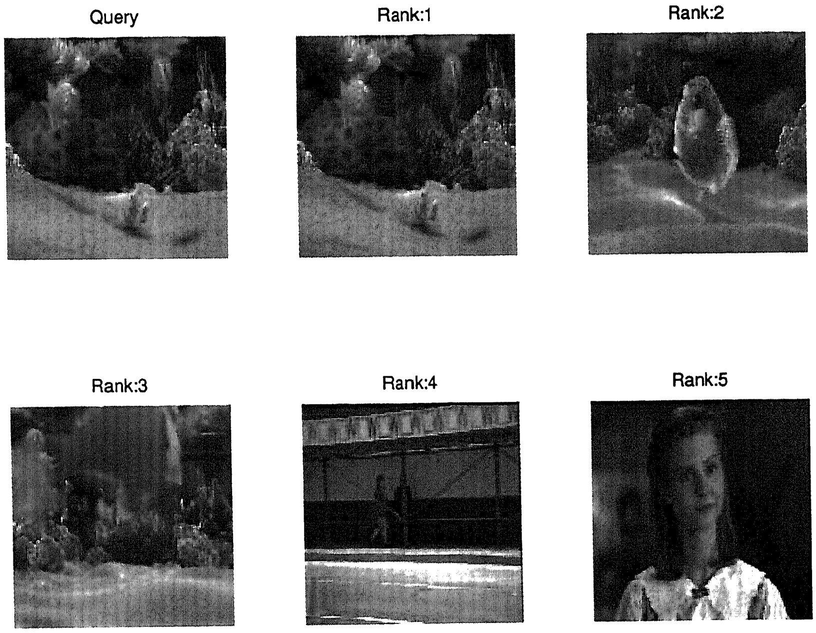


Figure 6.14: Query2:Retrieval in YC domain

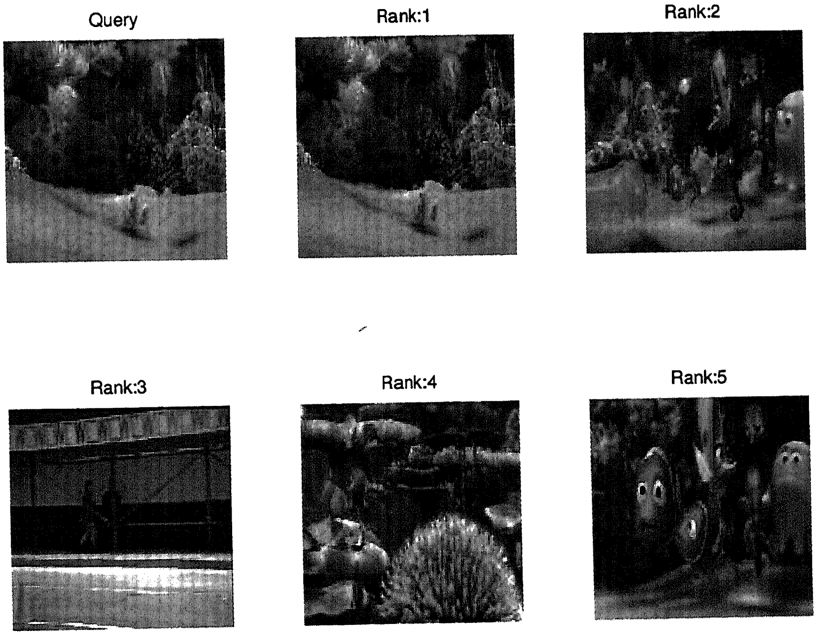


Figure 6.15: Query2:Retrieval in YCbCr domain

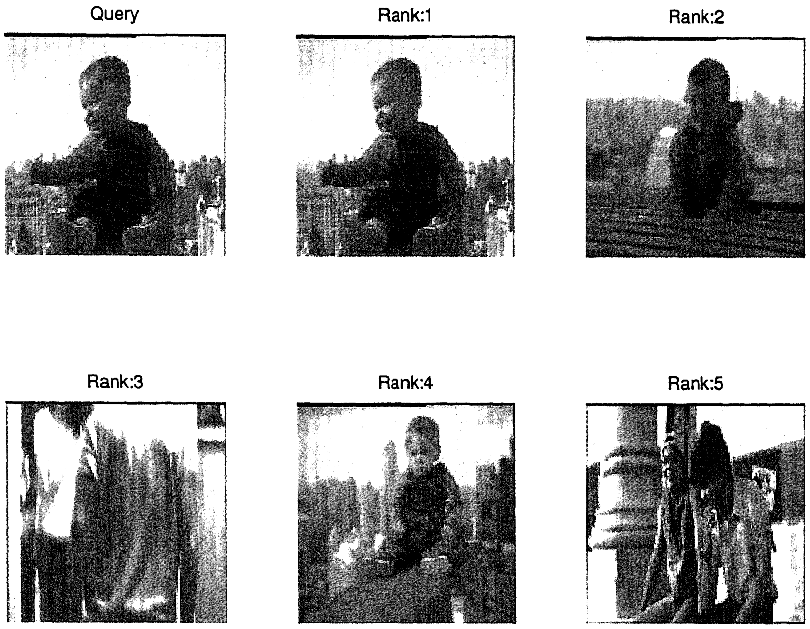


Figure 6.16: Query3:Retrieval in YC domain

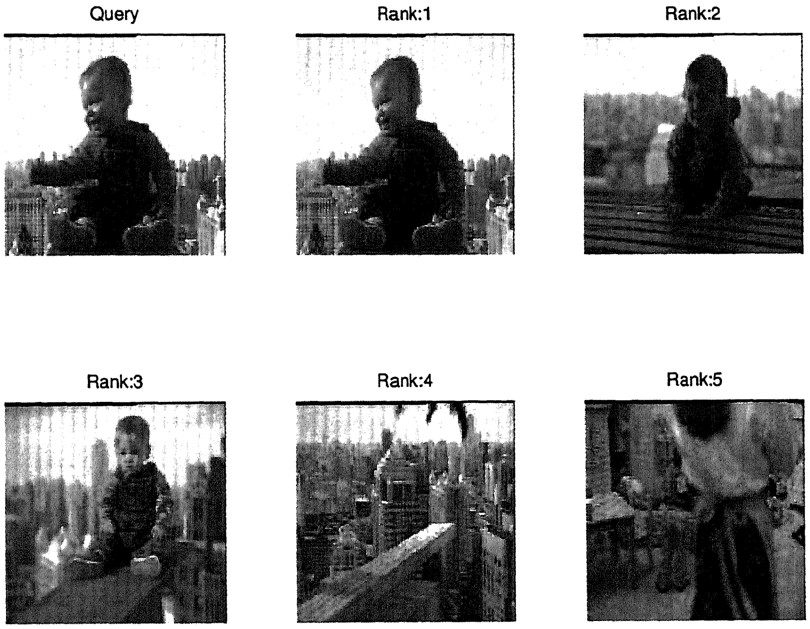


Figure 6.17: Query3:Retrieval in YCbCr domain

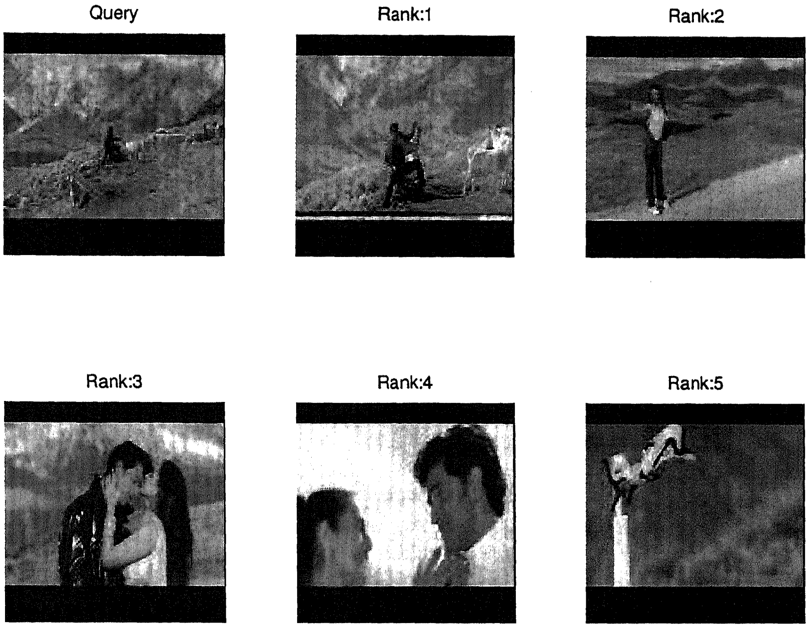


Figure 6.18: Query4:Retrieval in YC domain

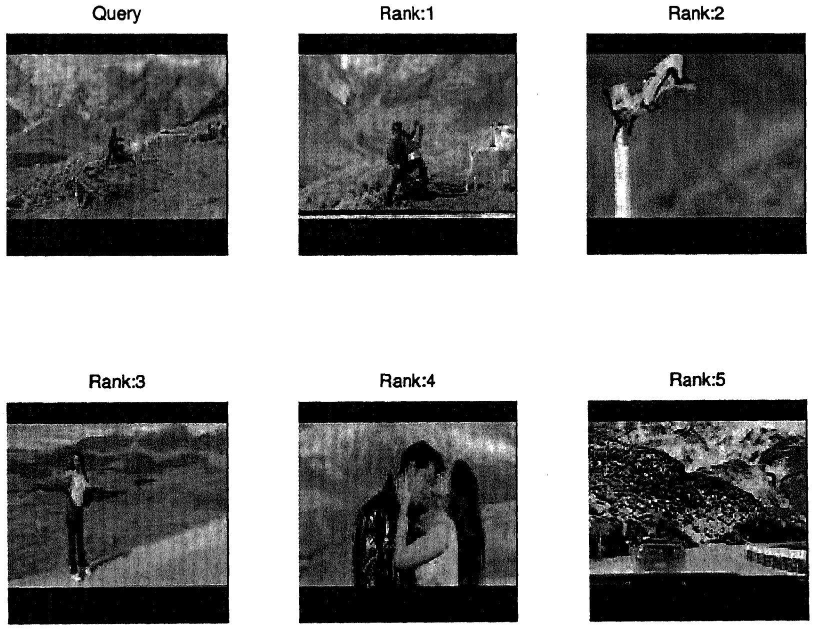


Figure 6.19: Query4:Retrieval in YCbCr domain

6.6 Summary of Video Indexing and Retrieval

In this section we give a summary of various steps involved in indexing and retrieval, and list the values of different parameters used in this thesis.

6.6.1 Various steps in video indexing

The following steps are involved in video indexing:

Step1 : The extraction of DC images of MPEG video.

Step2 : The scene change detection. All the DC images are mapped to YC domain. For spiral mapping, the value of $L = 7$ is used, where L represents the number of encirclements of spiral. Values of parameters for detecting abrupt scene changes are $m = 10$ and $n = 3$, where m and n are window length and multiplying factor respectively. The parameter F, which represents, the number of frames in dissolve region is a variable, and the values used are 10 for music videos, and 15 for other videos.

Step3 : Keyframe extraction of all the shots. Frames with minimum value of kurtosis are taken as keyframes.

Step4 : Keyframe clustering for video indexing. The values of threshold(ϵ) is taken as one. Lower ϵ is used to avoid false alarms in keyframe clustering.

Step5 : Indexing of keyframes for video indexing. The indices are stored in the database along with the keyframes. All the above steps are followed to index the videos in the database.

6.6.2 Steps in video retrieval

When a query is posed to the system indices of the query are extracted. These indices are compared to the indices of the keyframes stored in the database. Frames that have least distance from the query are retrieved.

Chapter 7

Conclusion and Scope for Future Work

7.1 Conclusions

In this thesis we proposed a novel video indexing and retrieval system based on two-dimensional representation of color. Mapping of color from 3D color representation(RGB) to 2D color representation(YC) is discussed. Experiments conducted using spiral mapping, indicate that it is a valid representation. The SNR calculations showed that for spiral approximation, 7 encirclements of the spiral give a good representation of image in YC color space. Scene change detection algorithm is modified such that both abrupt and gradual scene changes can be detected using the same data. An algorithm for keyframe clustering is proposed. This algorithm efficiently reduces the redundancies in the keyframe database. Color layout descriptor is used for indexing the keyframes and retrieval of video segments. Experimental results show that indexing in YC domain has the advantage of less storage cost. If the indices are coded, each descriptor in YC domain takes 47 bits, while each descriptor in YCbCr domain takes 63 bits. So there is a reduction of 16 bits per descriptor in YC domain. Retrieval time in YC domain is less than retrieval time in YCbCr domain. For a database of 936 keyframes, the reduction in retrieval time is found to be 14.46%. The results obtained confirm the validity of YC representation for video indexing and retrieval.

Bibliography

[1] Ilan Govan, 'Novel color TV system based on spiral sampling', *M.Tech. Thesis, Indian Institute of Technology Kanpur*, aug.1994.

[2] Mufit Ferman.A, Murat Tekalp.A, and Rajiv Mehrotra, 'Robust color histogram descriptors for video segment retrieval and identification', *IEEE transactions on Image Processing'*, vol.11, no.5, pp.497–508, may.2002.

[3] Kasutani.E, Yamada.E, 'The MPEG-7 Color Layout Descriptor: A Compact Image Feature Description for High-Speed Image/Video Segment Retrieval', *International Conference on Image Processing*, vol.1, pp.674–677, oct.2001.

[4] Eung Kwan Kang, Sung Joo Kim, and Jong soo Choi, 'Video retrieval based on keyframe extraction in compressed domain', *International Conference on Image Processing*, Vol. 3, pp.24–28, oct.1999.

[5] Thomas Sikora, 'The MPEG-7 visual standard for content description-an overview', *IEEE transactions on Circuits and Systems for Video Technology*, vol. 11, no. 6, pp.696–702, june.2001.

[6] Zhang.H.J, Kankanhalli.A, Smoliar.S.W, 'Automatic partitioning of full-motion video', *Multimedia Systems,* vol.1, no.1, pp.10–28, 1993.

[7] Kasturi.R and Jain.R, 'Dynamic vision' in Kasturi.R and Jain.R (eds.): *Computer Vision: Principles*, IEEE Computer society press, Washington, pp.469–480, 1991.

[8] Nagasaka.A and Tanaka.Y, ‘Automatic video indexing and full-video search for object appearances’ in Kunth.E and Wegner.L.M.(eds): *Visual Database Systems II*, Elsevier Science, pp.113–127, 1992.

[9] Zabith.R, Miller.J, and Mai.K, ‘A feature-based algorithm for detecting and classifying production effects’, *Multimedia Systems*, vol.7, no.2, pp.119–128, 1999.

[10] Arman.R, Hsu.A, and Chiu.M.Y, ‘Image processing on compressed data for large video database’, *First ACM International Conference on Multimedia*, pp.267–272, aug.1993.

[11] Zhang.H.J, Low.C.Y, and Smoliar.S.W, ‘Video parsing and browsing using compressed Data’, *Multimedia Tools and Applications*, vol.1, pp.89–111, 1995.

[12] Yeo.B.L and Liu.B, ‘On the extraction of dc sequences from MPEG compressed video’, *International Conference on Image Processing*, Vol. 2, pp.23–26, oct.1995.

[13] Fernando.W.A.C, Canagarajah.C.N and Bull.D.R, ‘Scene change detection algorithms for content based video indexing and retrieval’ *Electronic and Communication Engineering Journal*, pp.117–125, june.2001.

[14] Yeo.B.L and Liu.B, ‘Rapid scene analysis on compressed video’, *IEEE transactions on Circuits and Systems for Video Technology*, vol.5, no.6, pp.532–544, dec.1995.

[15] Diklic.D, Petkovic.D, Danielson.R, ‘Automatic extraction of representative keyframes based on scene content’ *Thirty-Second Asilomar Conference on Signals, Systems and Computers*, vol.1, pp.877–881, 1998.

[16] Saraceno.R, Leonardi.R, ‘Identification of visual correlations between non consecutive shots in digital image sequence’ *VLBV 98, Beckman Institute for Advanced Sciences and Technology, University of Illinois, Urbana-Champaign*, pp.65–68, 1998.